Project Documentation

TITLE

"Understanding User Sentiments in App Reviews: Descriptive Analysis and Predictive Modeling"

Descriptive Analysis Objective

Objective: Conduct sentiment analysis on user reviews of apps to understand user sentiment and satisfaction levels.

Methodology

1. Data Import and Cleaning

- Imported data for app1 and app2.
- Removed duplicate data present in both datasets.

2. Data Preparation

- Added an identifier column to differentiate app1 and app2 after concatenation.
- Concatenated data from both apps for further analysis.

3. Sentiment Analysis

- Analyzed over 1.5 lakh user reviews for the two apps using unsupervised methods such as TextBlob and VADER.
- Classified reviews into positive, negative, and neutral sentiments.

4. Insights into User Feedback

- Examined sentiment distribution and reviewed word/character counts to gain insights.
- Analyzed the length of characters and words for negative and positive reviews, and plotted histograms.

- Plotted pairplots and heatmaps to visualize the correlation between different attributes.

5. Visualizations

- Generated word clouds and bar charts for the top positive and negative reviews, including unigrams and bigrams.
- Separated the concatenated data back into app1 and app2.
- Drew inferences on user sentiments from the average length of words and characters in positive and negative sentiments.
- Generated word clouds and bar charts for top positive and negative reviews, including unigrams and bigrams, for both apps separately.

6. Comparison and Visualization

- Created a comparison table with columns for app name, and positive and negative sentiment distribution.
- Visualized sentiment distribution of both apps side by side using stacked bar plots and pie charts.

7. Conclusion

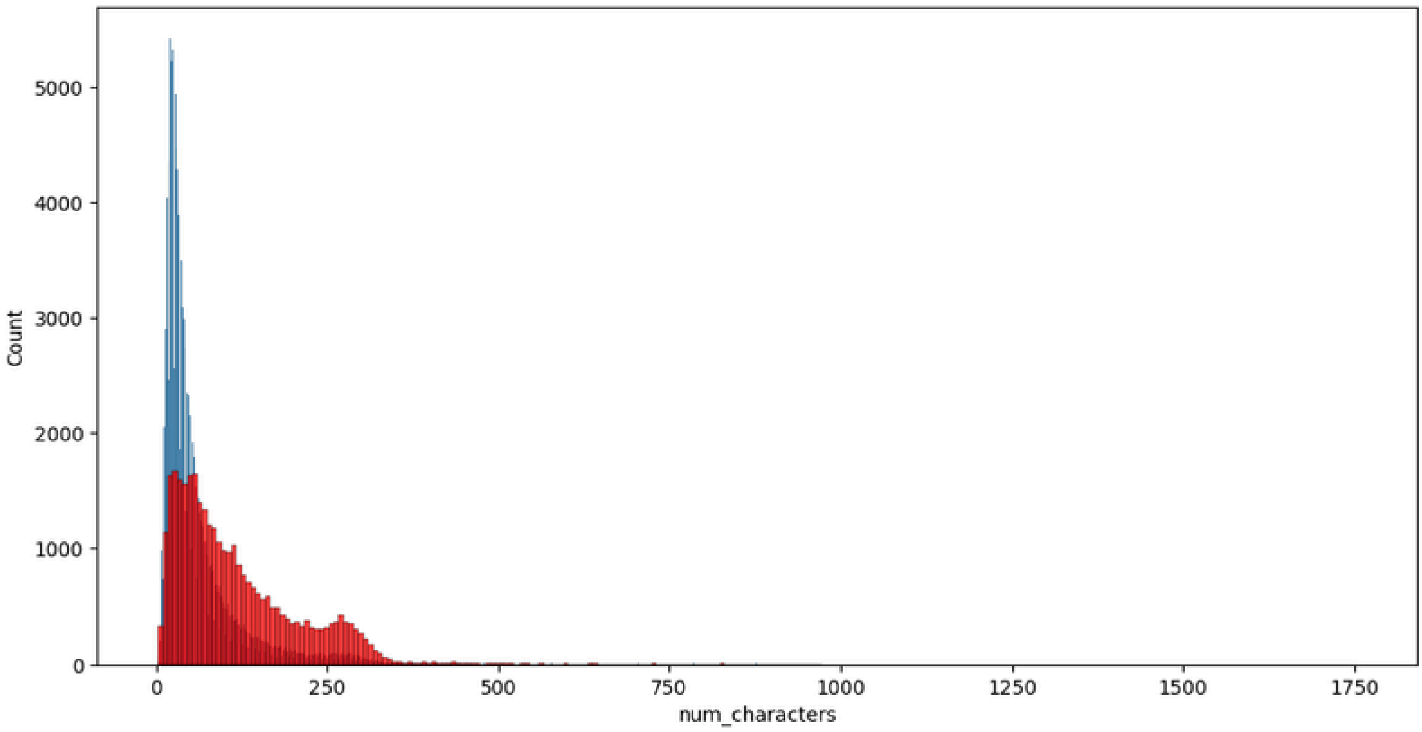
- Drew conclusions on which app is better based on all the above observations.

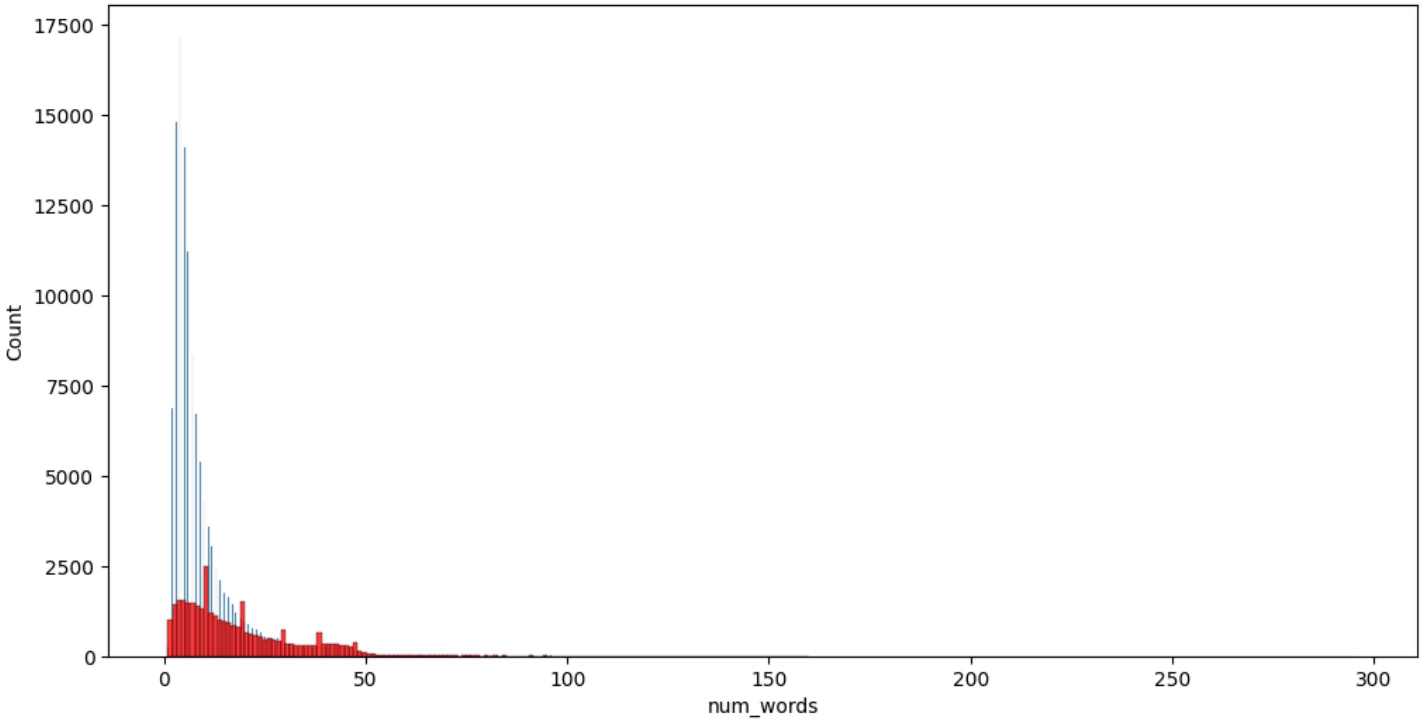
Observations

1. Length Analysis

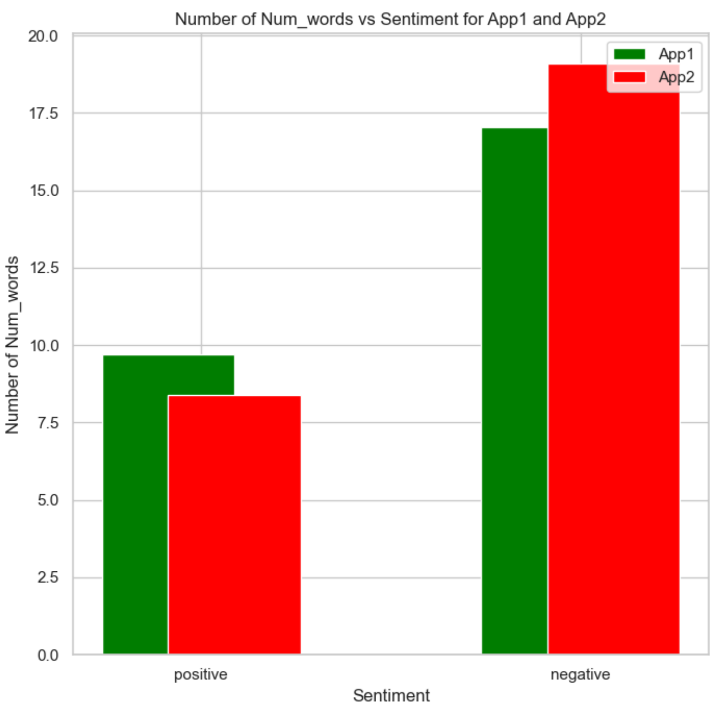
- The length of characters and words in negative reviews is greater than in positive reviews.

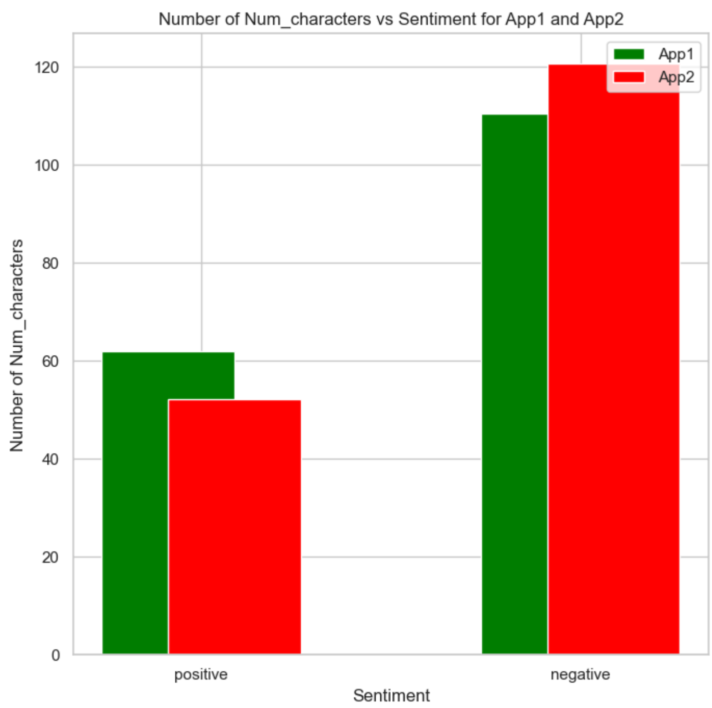
Combined Apps Data Plot:





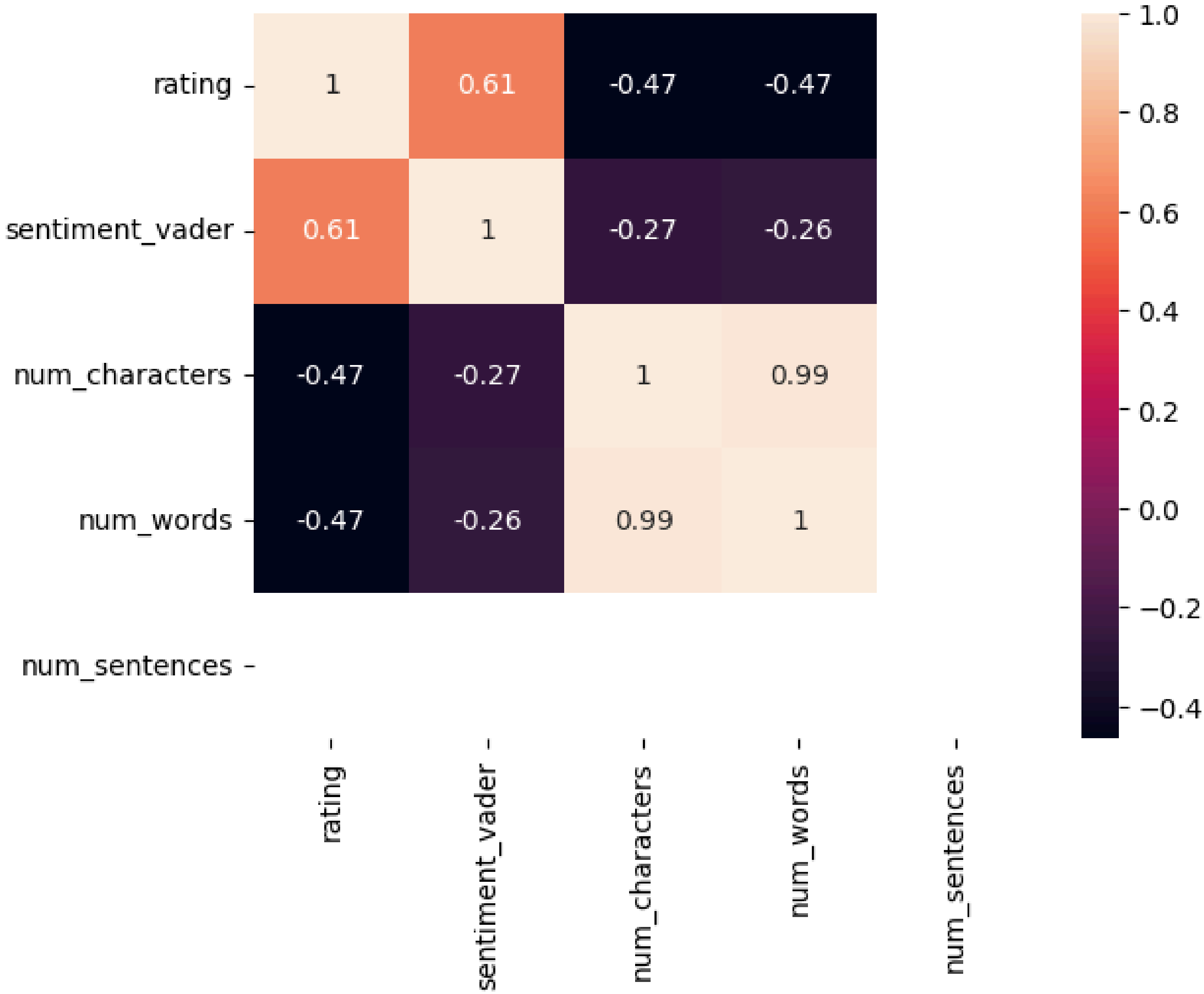
Separate Apps Data Mean Length of Words and Characters vs Sentiments Plot:





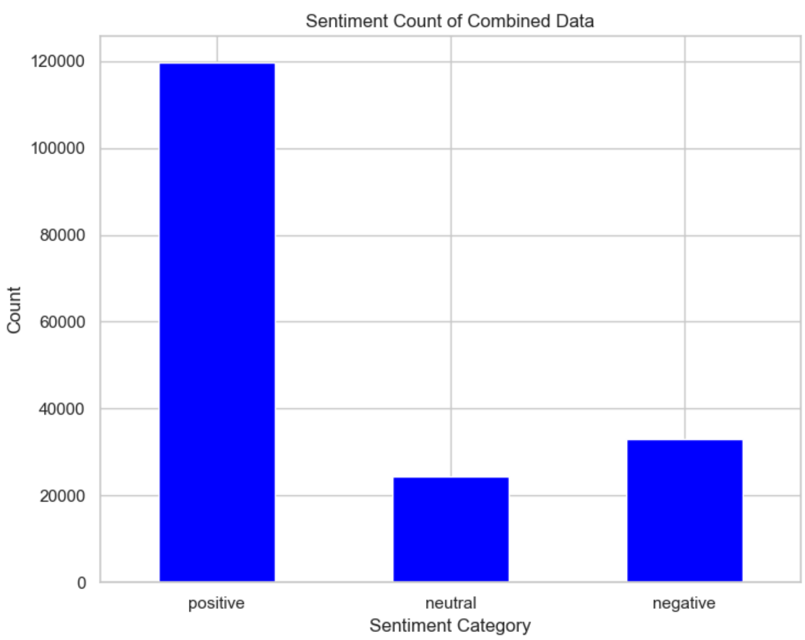
2. Correlation Insights

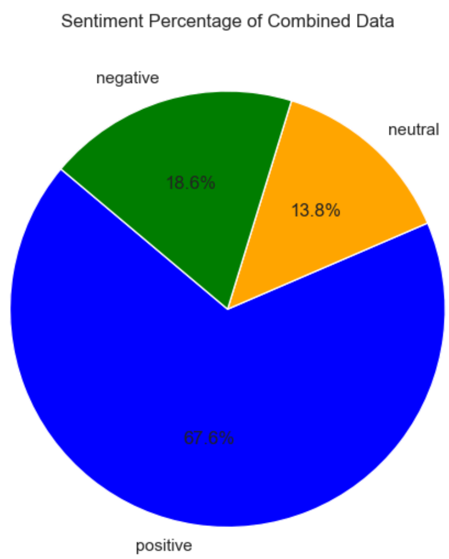
- There is a high correlation between the number of words and characters.
- Ratings and word/character counts are negatively correlated, indicating that reviews with a higher word count tend to have lower ratings/negative reviews.



3. Sentiment Distribution in Combined Data

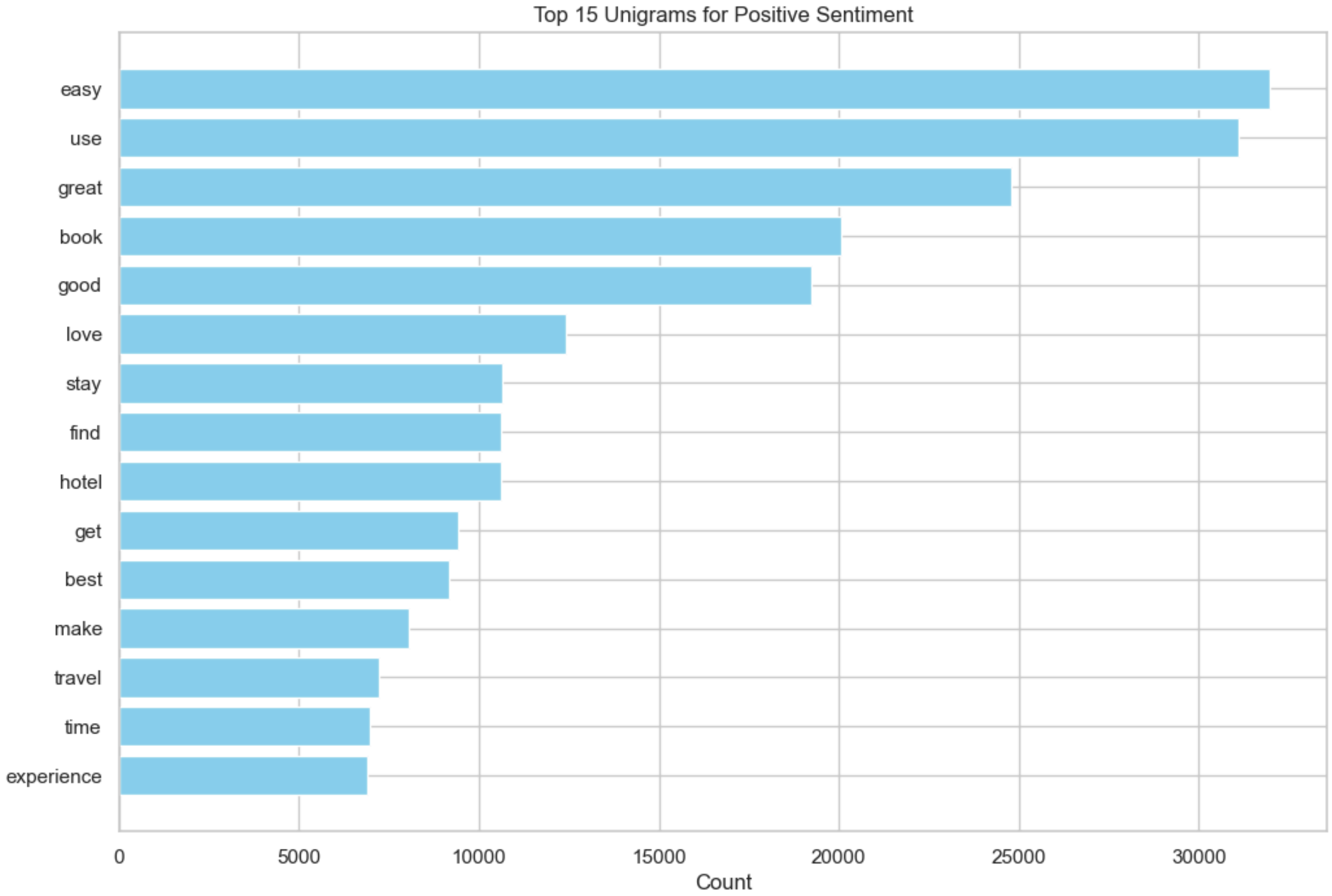
- Sentiment distribution in the combined app data is as follows:

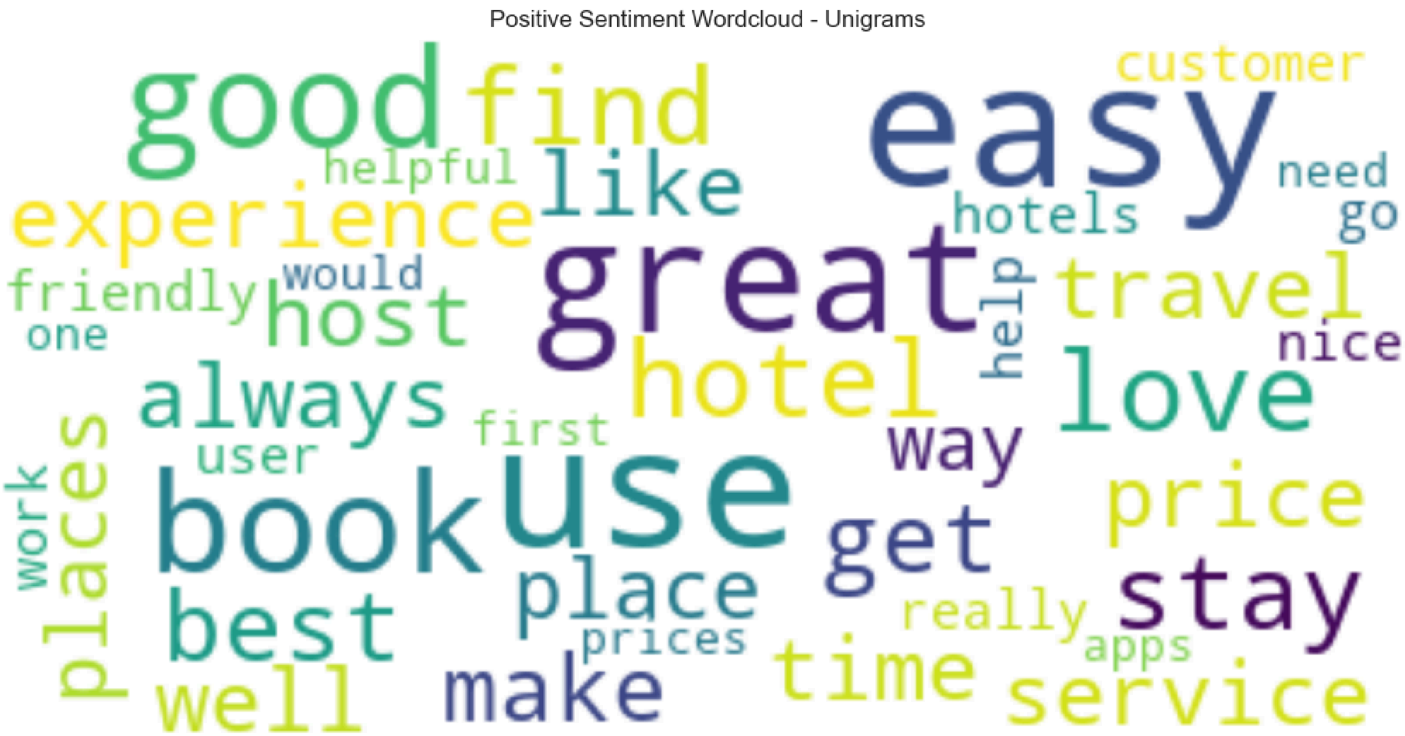




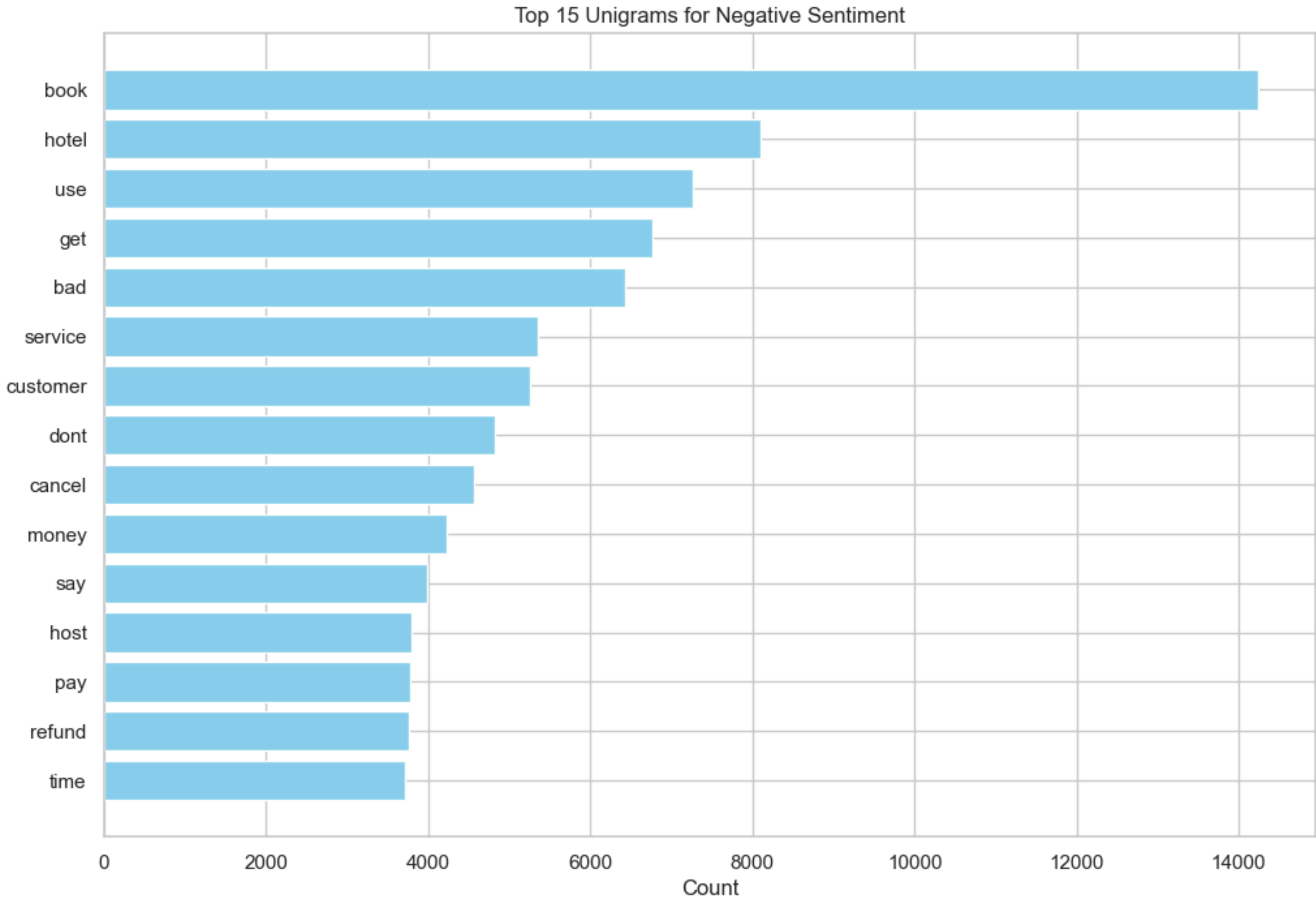
4. Top Unigrams and Bigrams in Combined Data

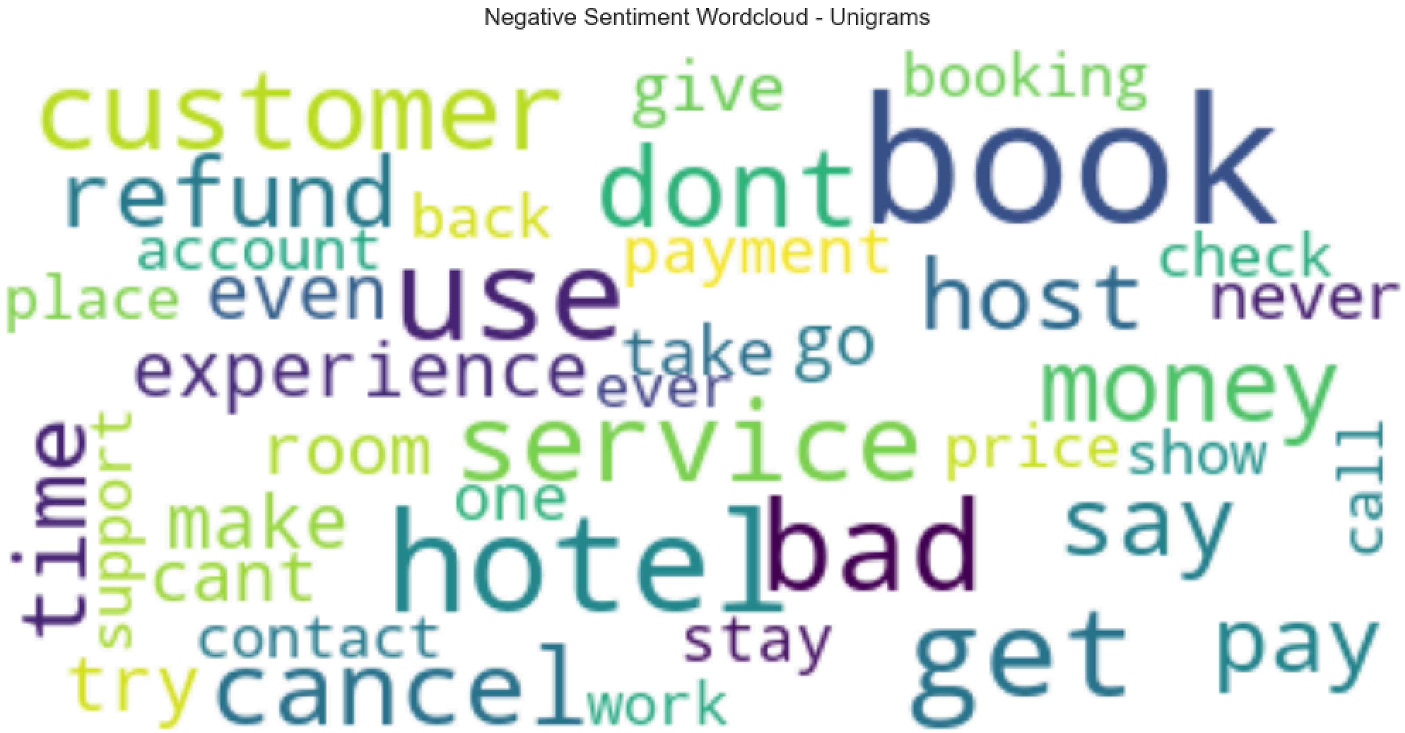
- The top unigrams and bigrams for positive and negative sentiments in the combined data were as follows:
- Positive Unigrams:



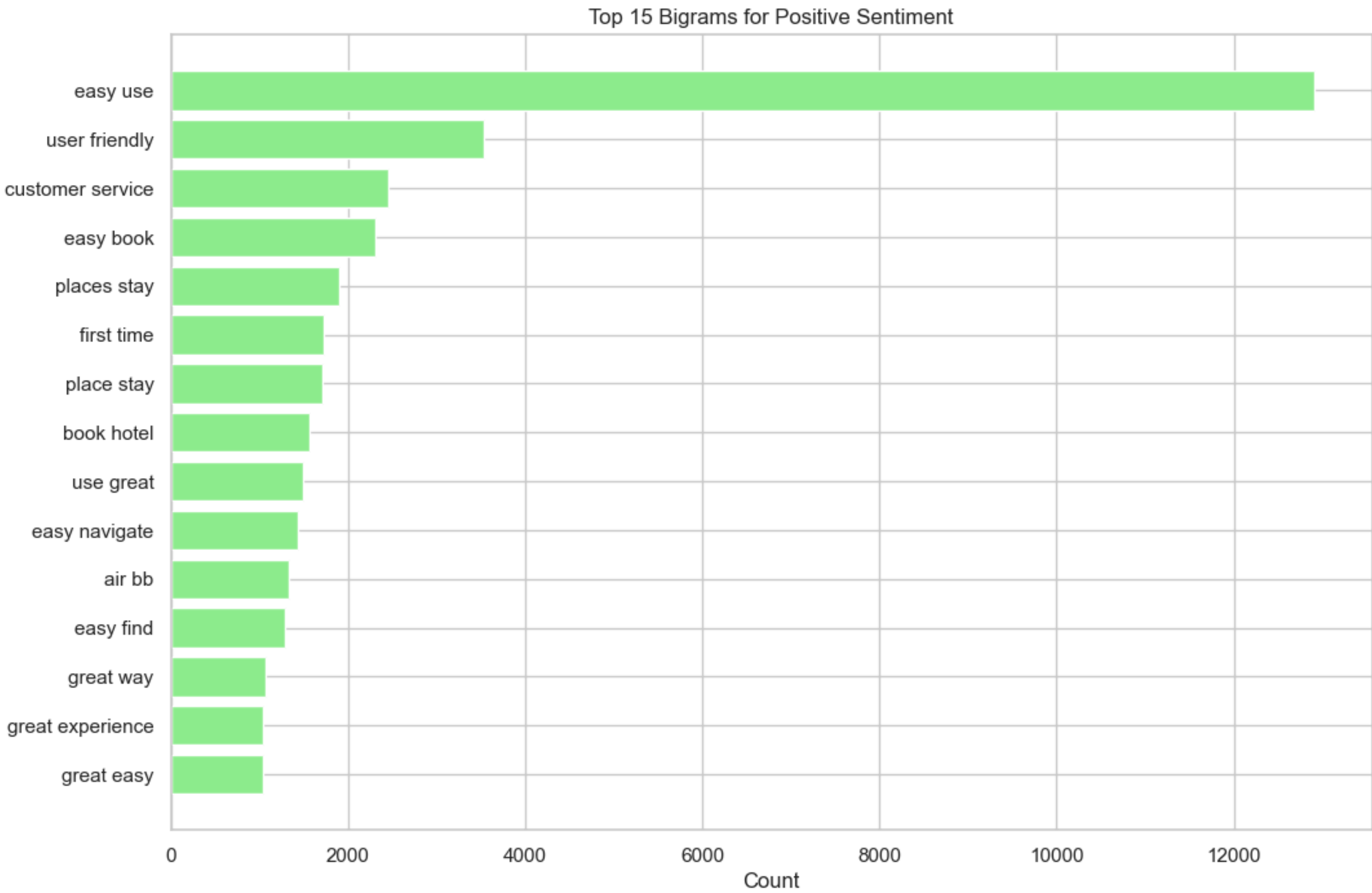


- Negative Unigrams:

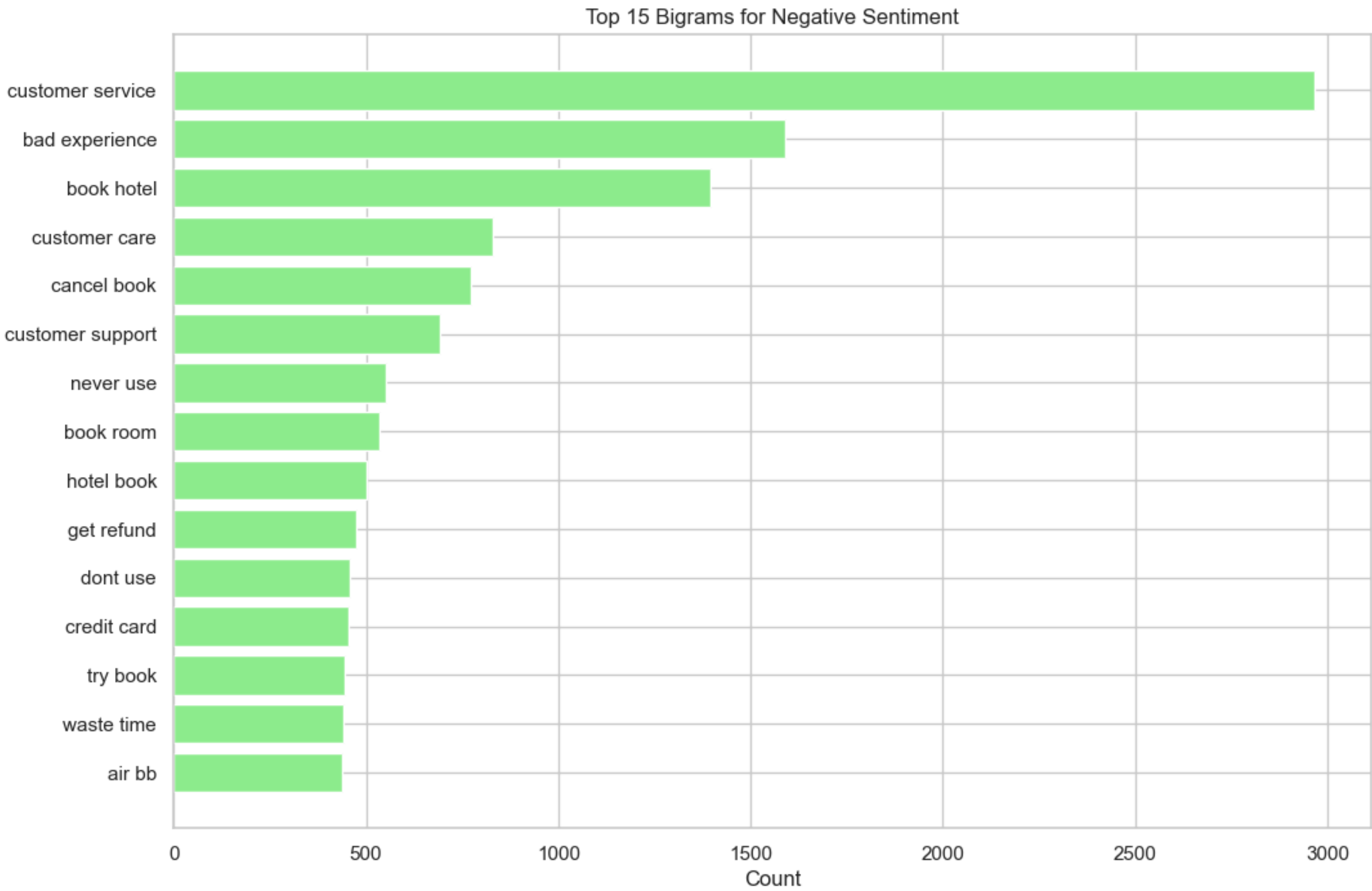




- Positive Bigrams:

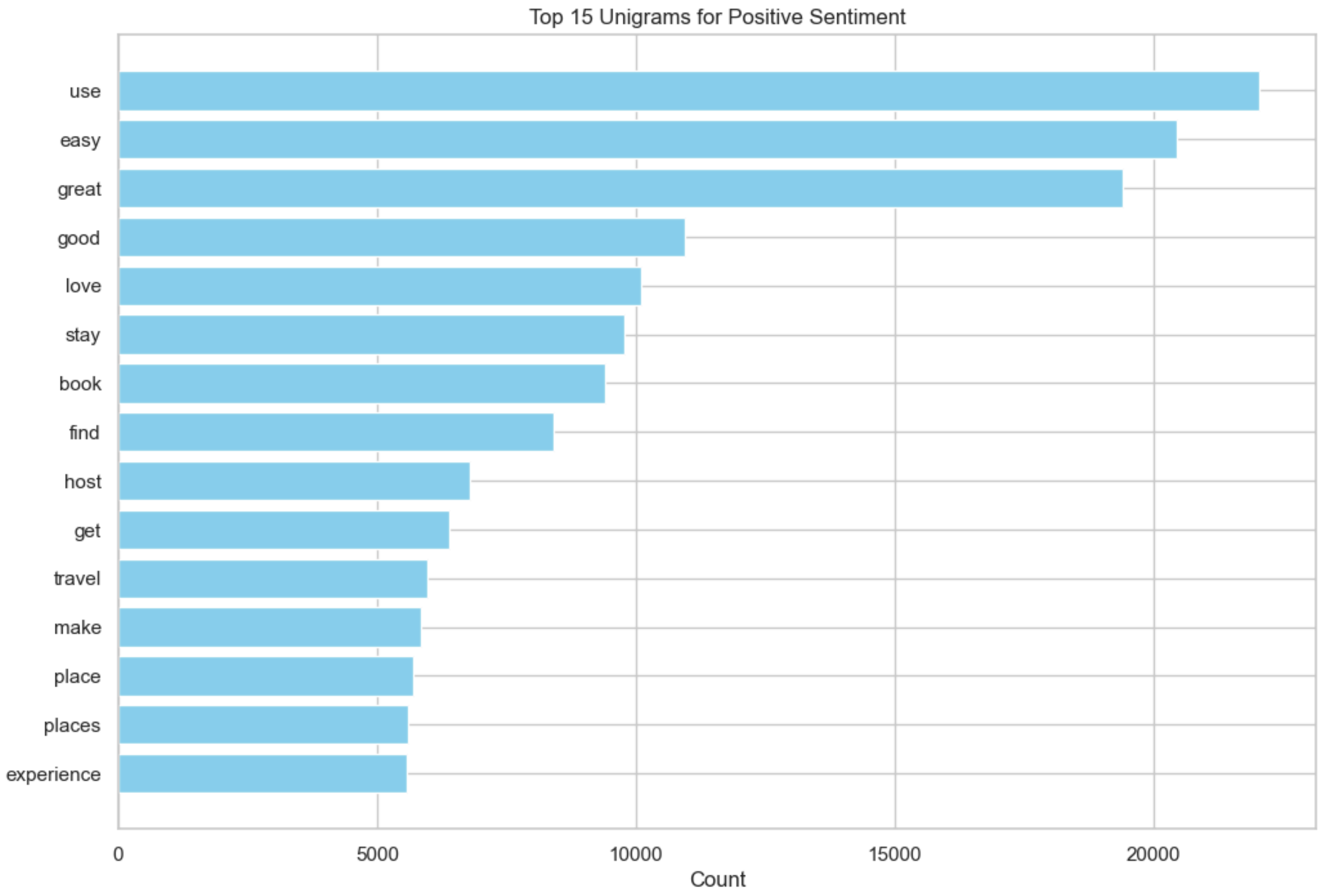


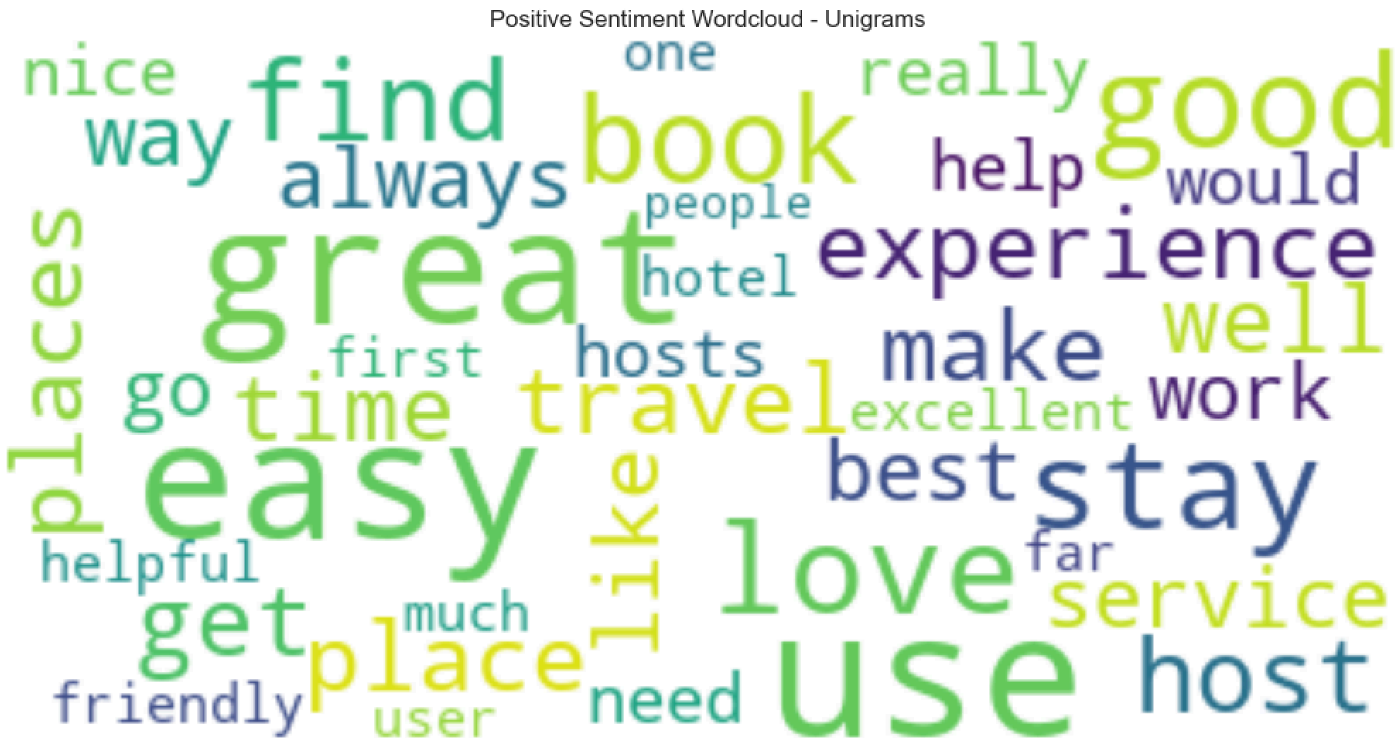
- Negative Bigrams:



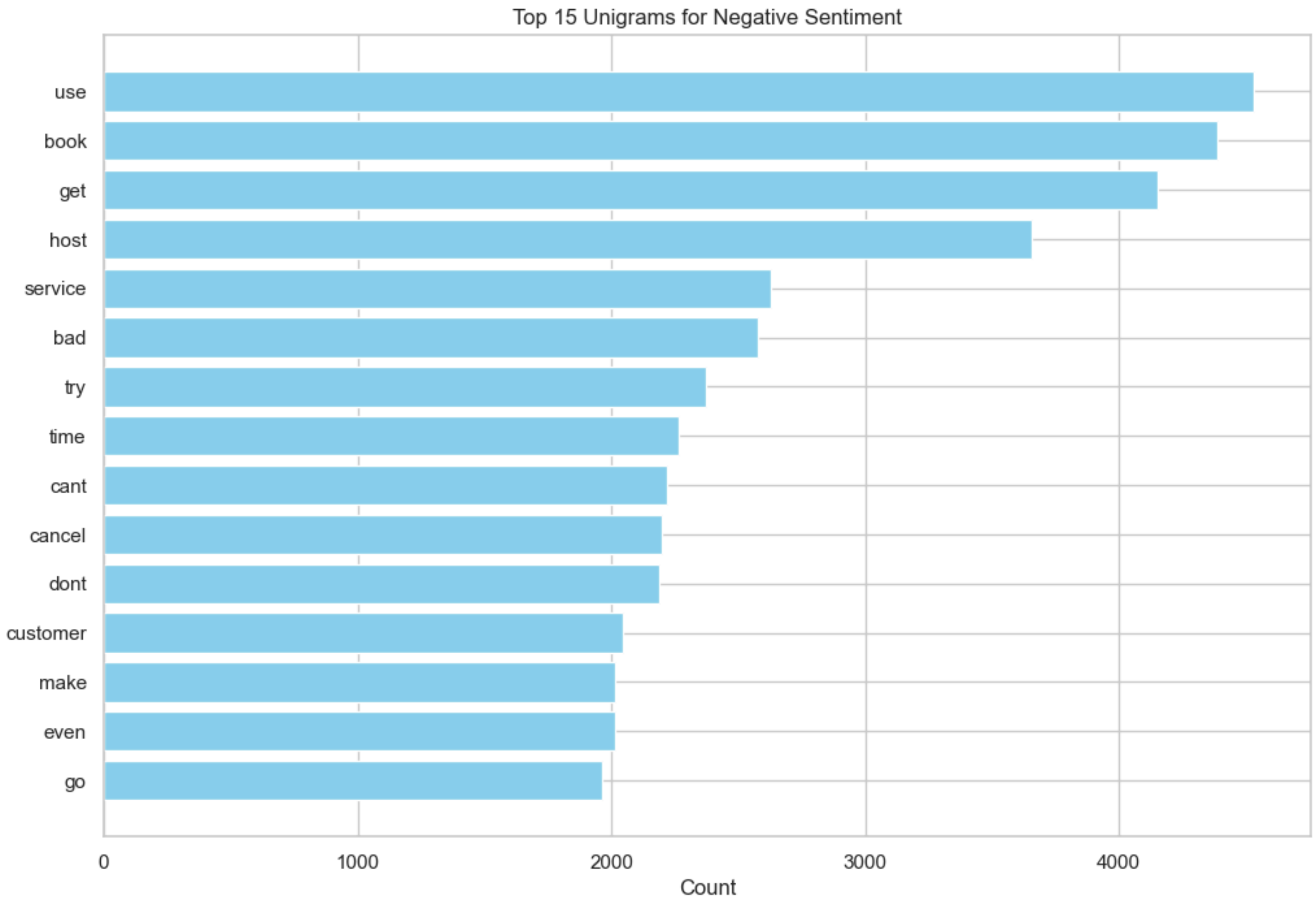
5. Top Unigrams and Bigrams in App1 Data

- The top unigrams and bigrams for positive and negative sentiments in app1 data were as follows:
- Positive Unigrams:

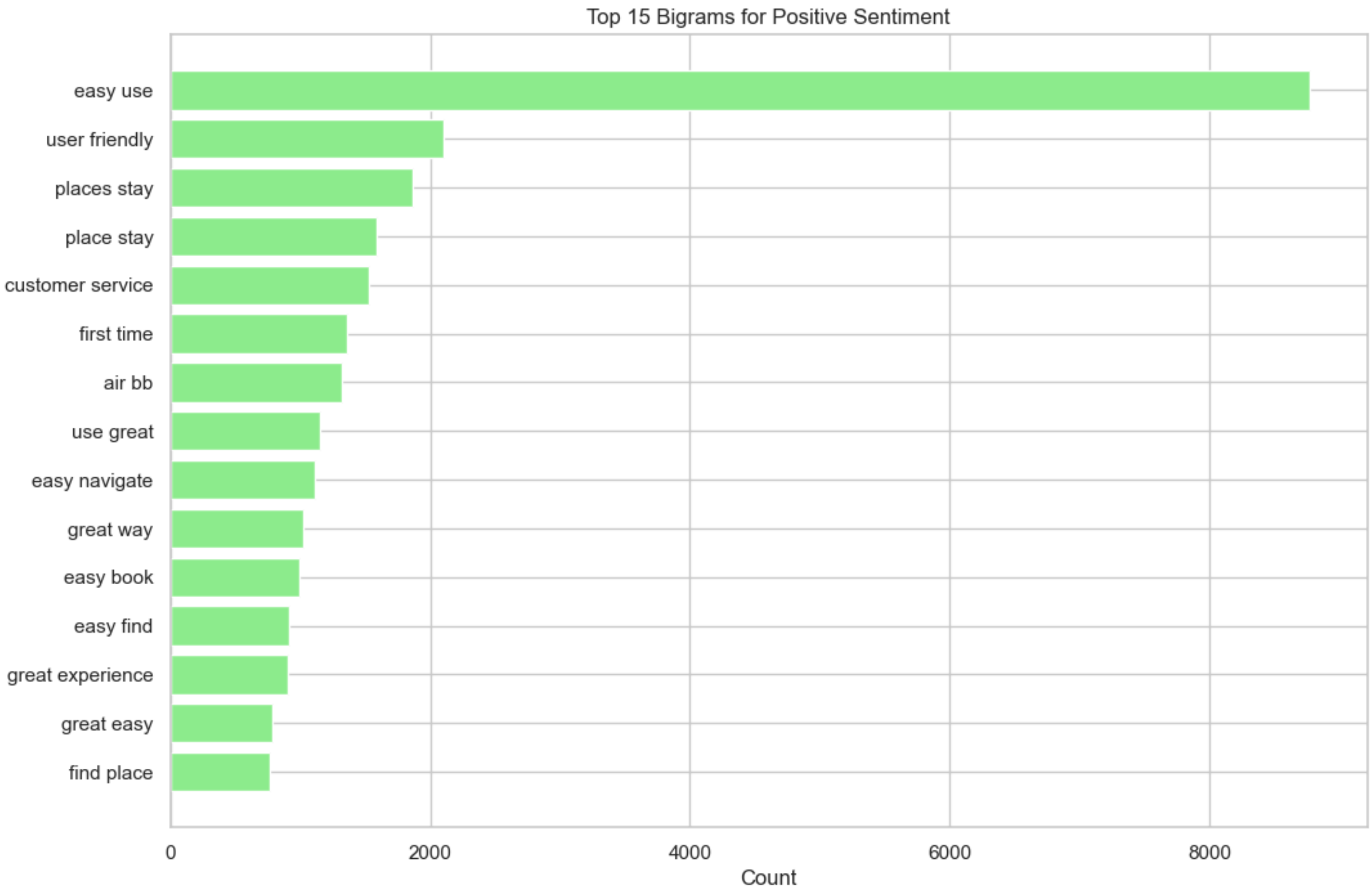


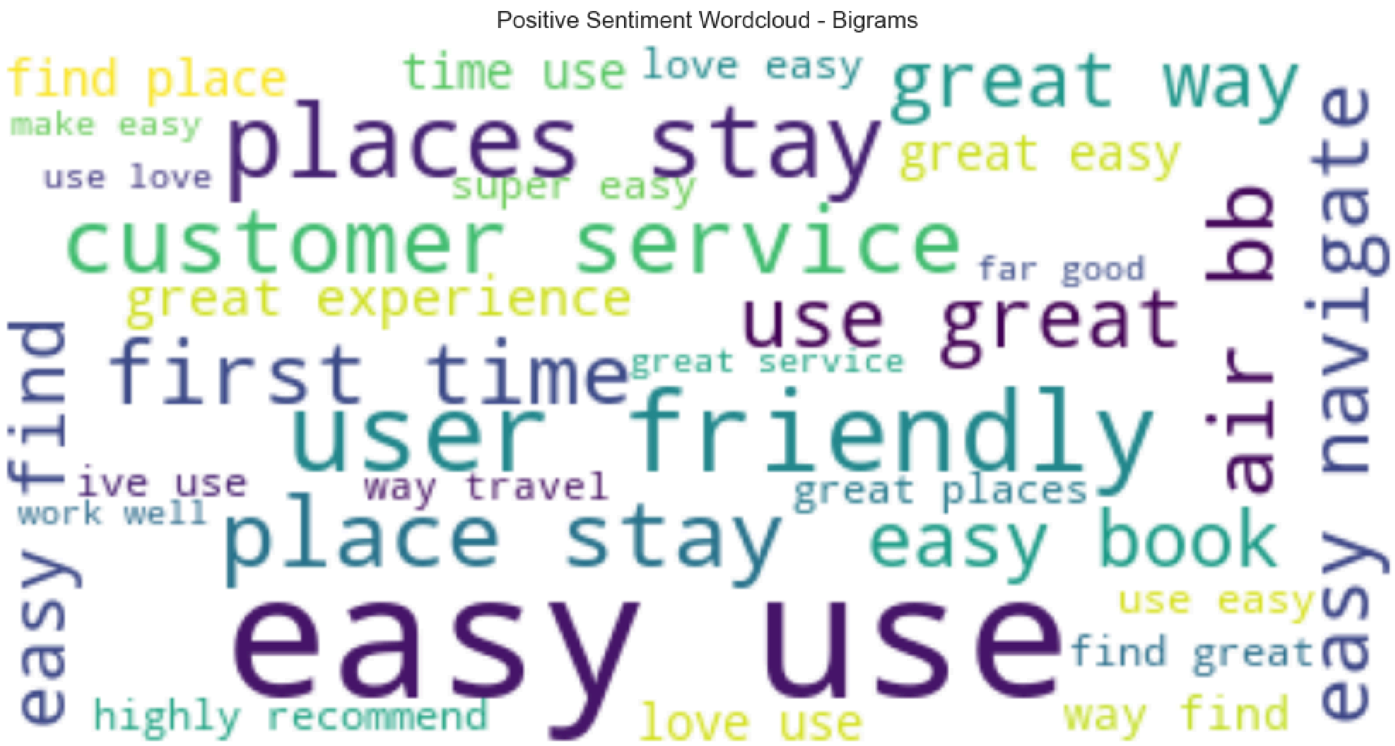


- Negative Unigrams:

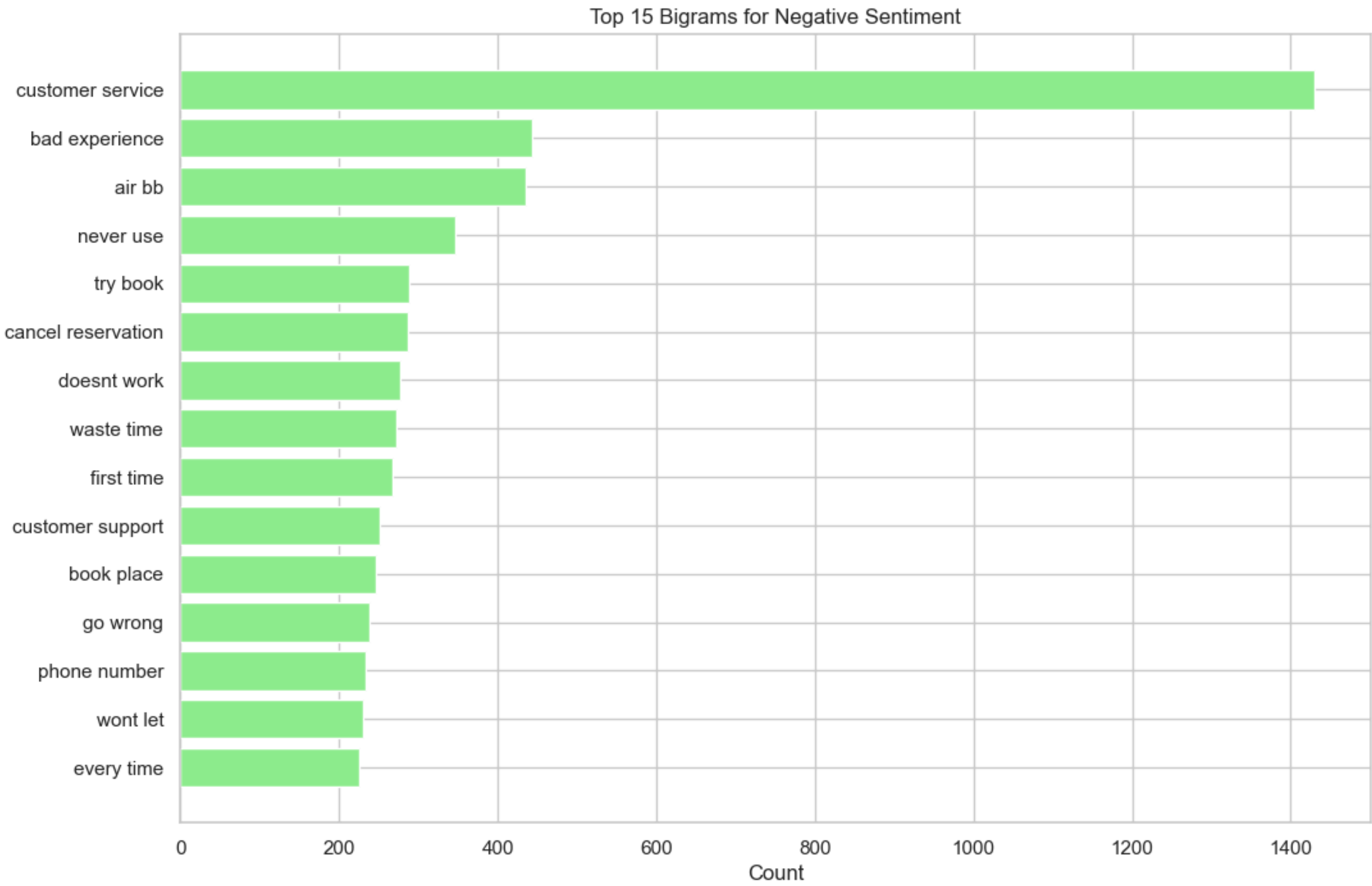


- Positive Bigrams:



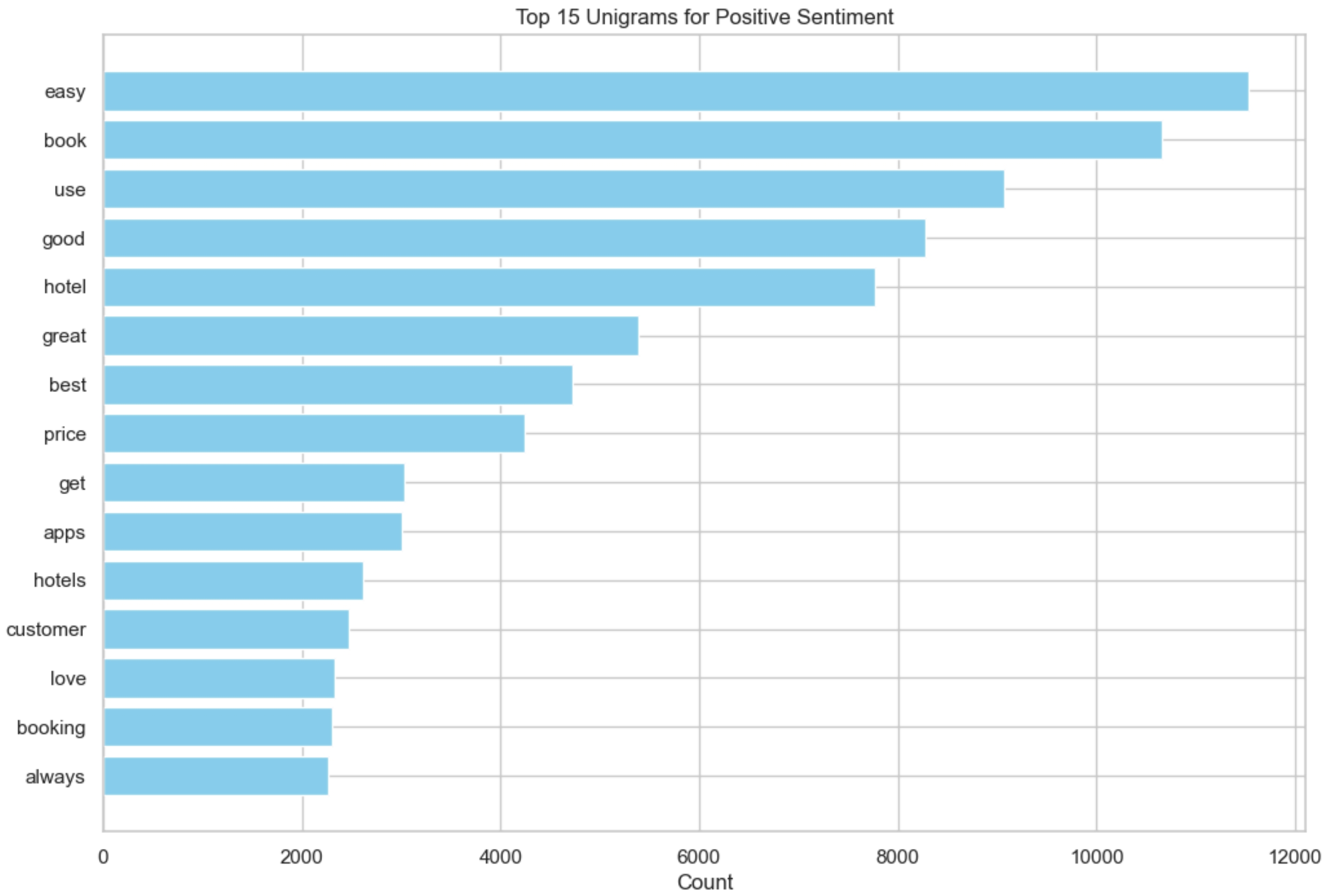


- Negative Bigrams:

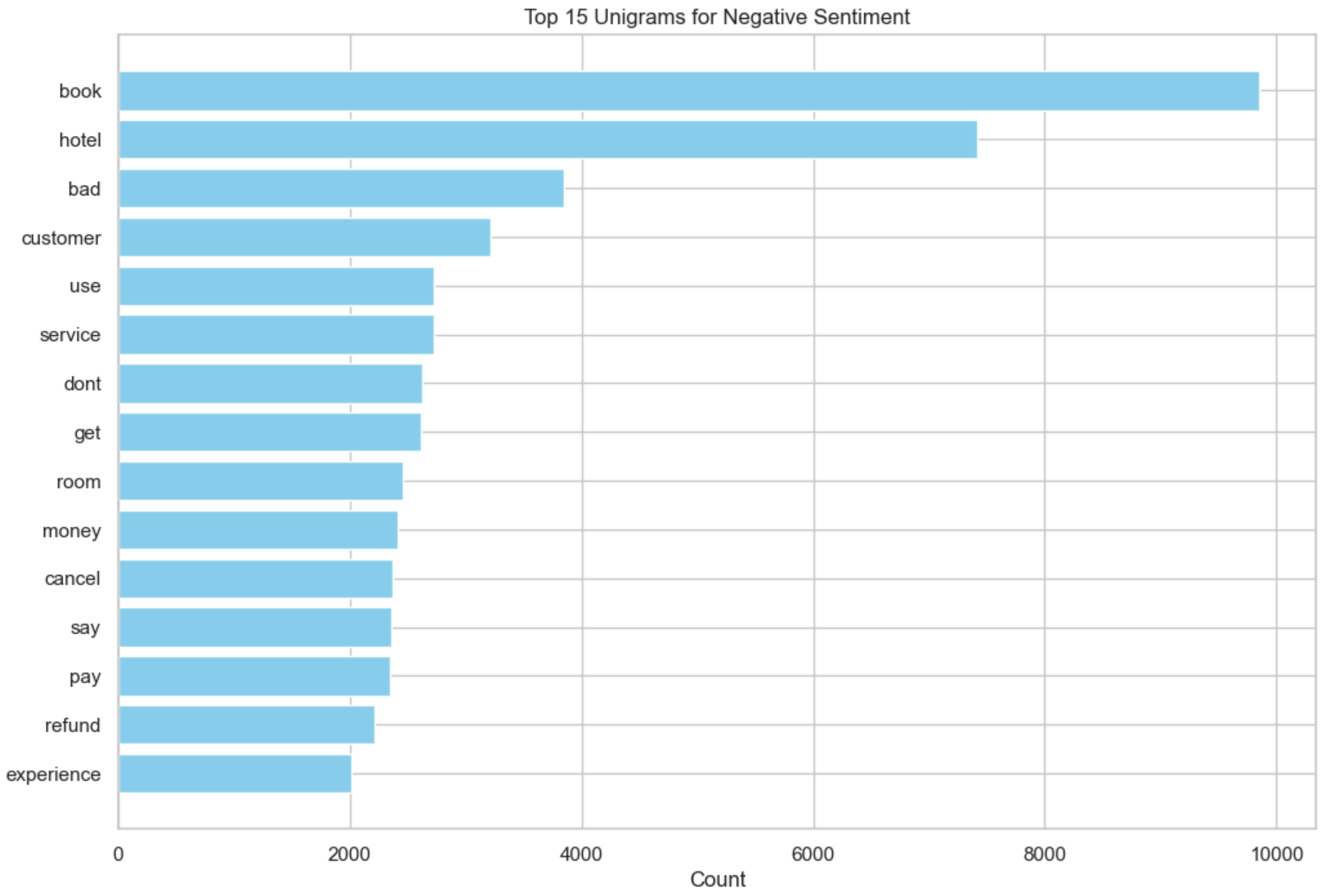


6. Top Unigrams and Bigrams in App2 Data

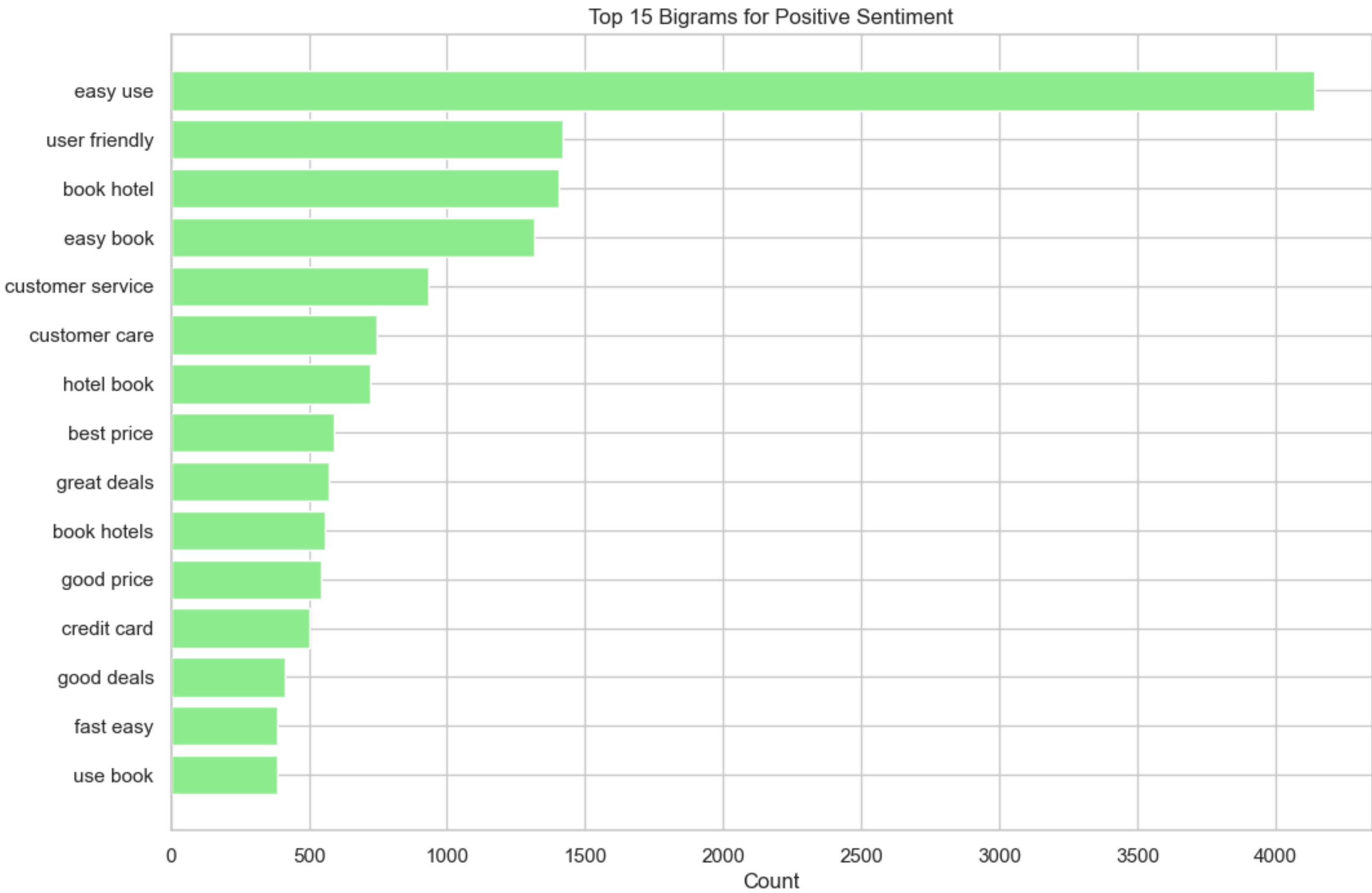
- The top unigrams and bigrams for positive and negative sentiments in app2 data were as follows:
- Positive Unigrams:



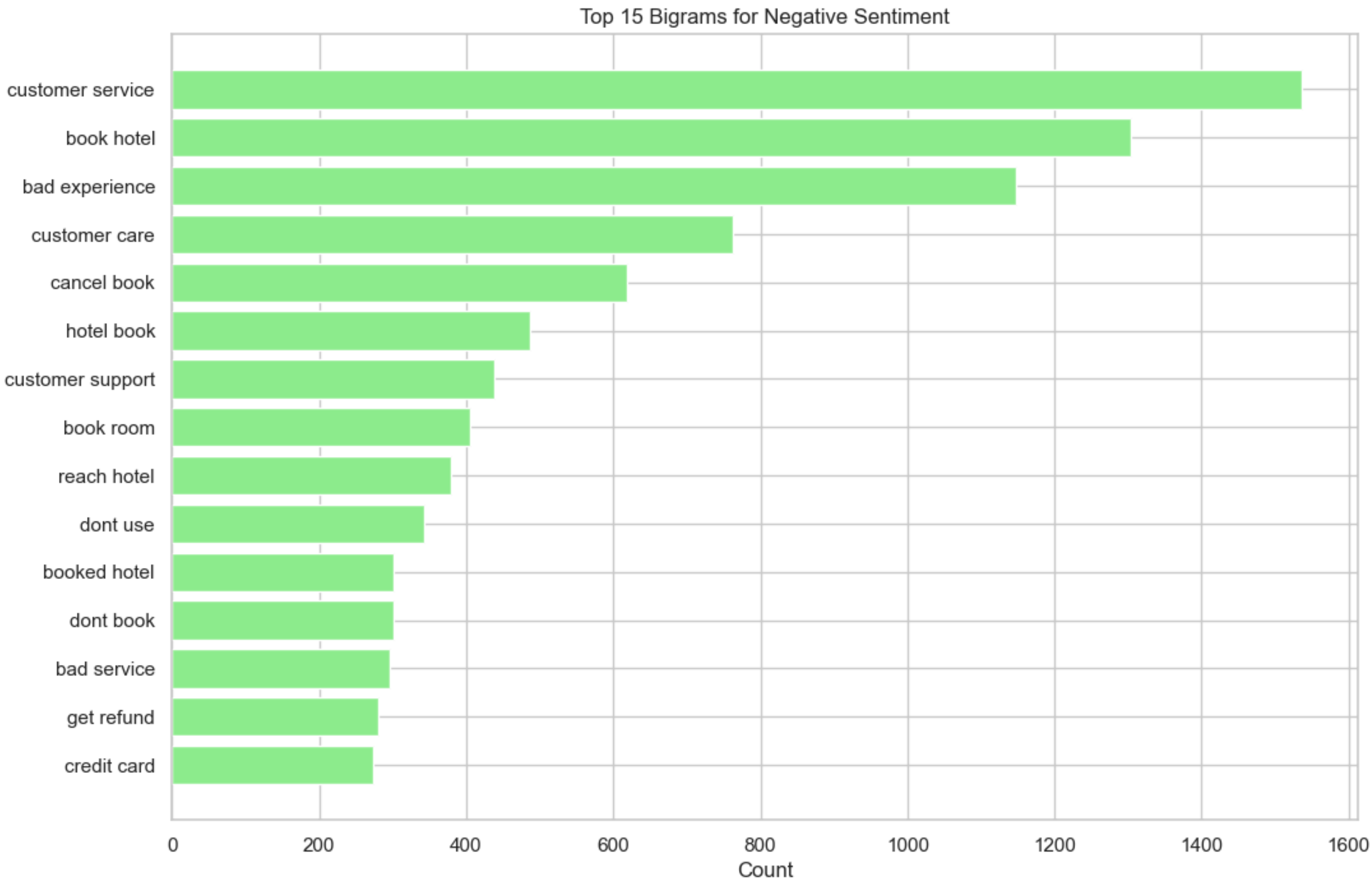
- Negative Unigrams:

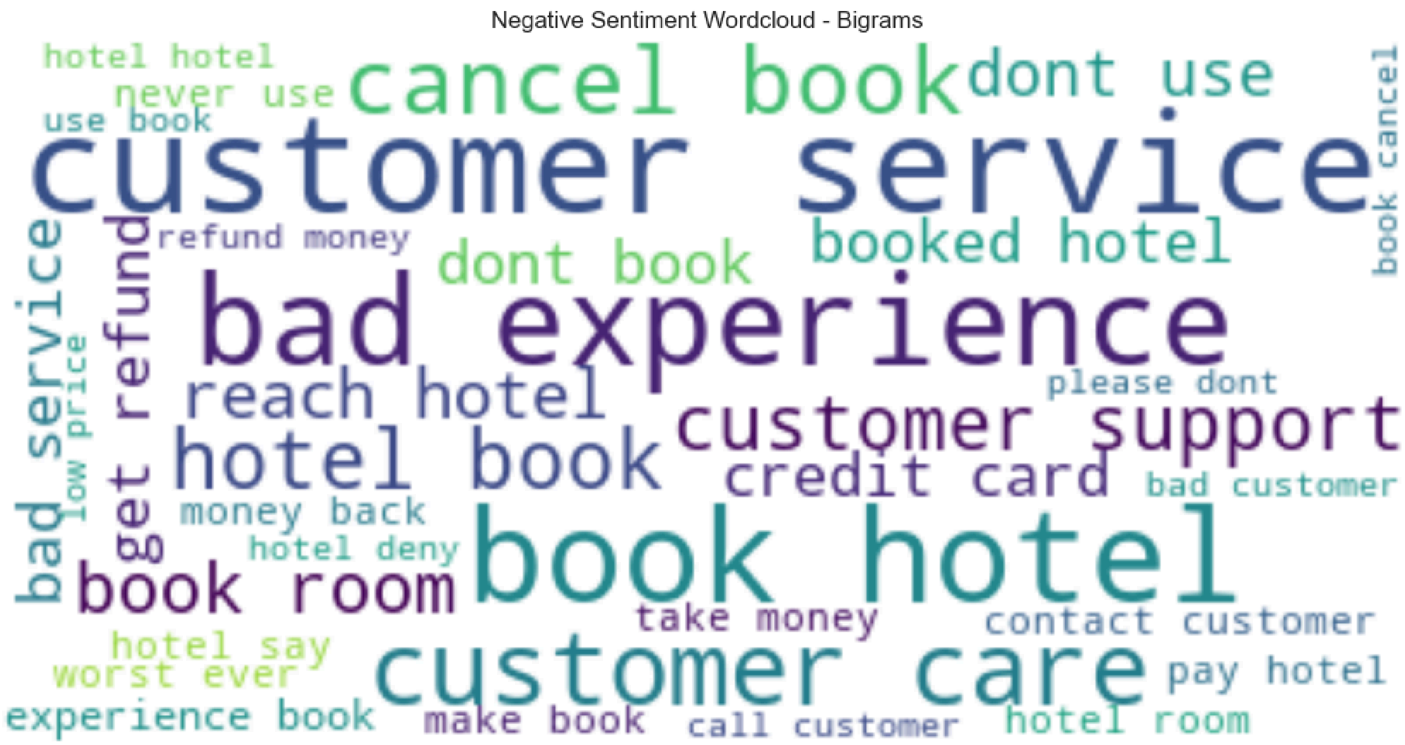


- Positive Bigrams:



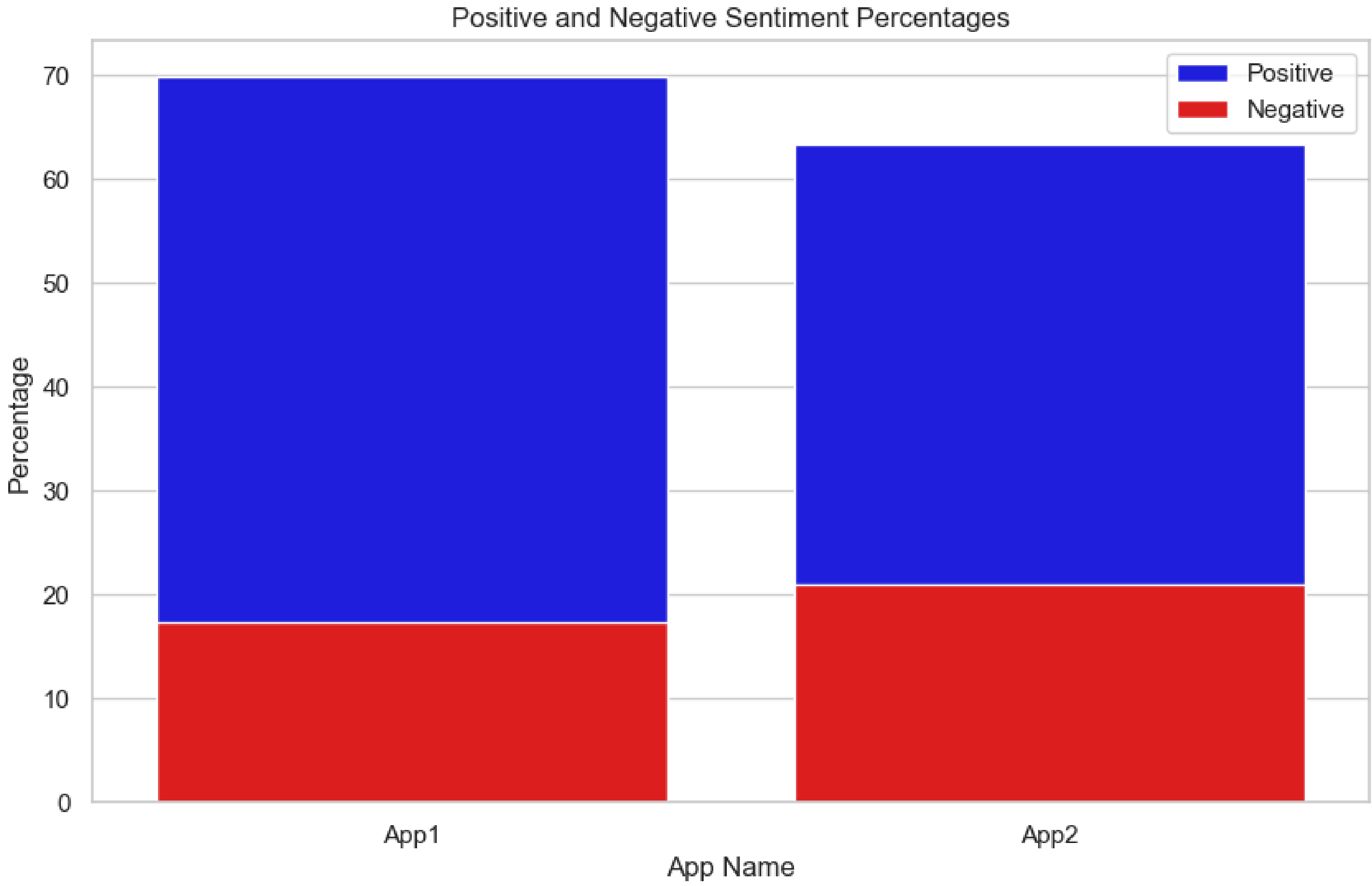
- Negative Bigrams:

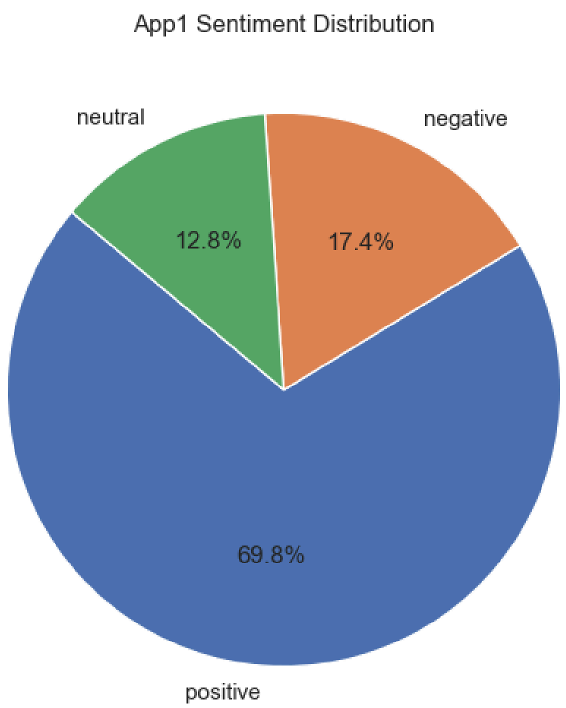


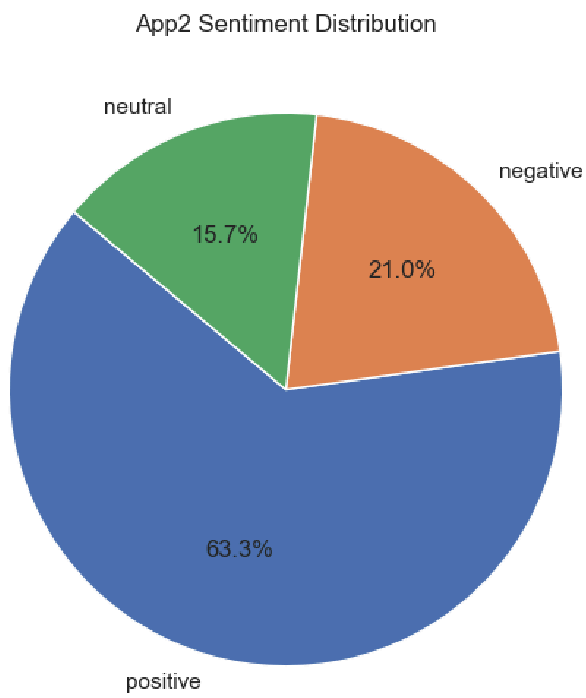


7. Sentiment Distribution in Both Apps

	app_name	totalSenti_count	positiveSenti_count	negativeSenti_count	positiveSenti_percentage	negativeSenti_percentage
0	App1	115354	80566	20029	69.842398	17.363074
1	App2	61746	39103	12971	63.328799	21.007029







INFERENCES

1. Positive Sentiments

- App1: 69.84%
- App2: 63.33%

- Conclusion: App1 has a higher percentage of positive sentiments by 6.51 percentage points, indicating that more users have a positive experience with App1 compared to App2.

2. Neutral Sentiments

- App1: 12.79%
- App2: 15.66%
- Conclusion: App2 has a higher percentage of neutral sentiments by 2.87 percentage points. This suggests that a slightly higher proportion of users feel indifferent or neutral about App2 compared to App1.

3. Negative Sentiments

- App1: 17.36%
- App2: 21.01%
- Conclusion: App1 has a lower percentage of negative sentiments by 3.65 percentage points, indicating that fewer users have a negative experience with App1 compared to App2.

4. Decision

- Higher Positive Sentiments: App1 has a significantly higher percentage of positive sentiments. This is a strong indicator of user satisfaction and preference.
- Lower Negative Sentiments: App1 also has a lower percentage of negative sentiments, suggesting fewer users have a poor experience with the app.
- Neutral Sentiments: While App2 has a higher percentage of neutral sentiments, this typically indicates a more indifferent user base. The higher positive and lower negative sentiments of App1 outweigh App2's neutral sentiments, as App1 has a higher percentage of positive sentiments by 6.51 percentage points, whereas App2 has a higher percentage of neutral sentiments by just 2.87 percentage points.

5. Conclusion

- App1 is likely the better app in terms of user satisfaction. It has a higher proportion of positive feedback and a lower proportion of negative feedback compared to App2.

Predictive Analysis Objective

Objective: Compare the performance of ML and DL algorithms in predicting sentiments based on trained models.

ML Methodology

1. Data Loading and Preparation

- Loaded saved concatenated data of both apps and removed duplicate rows.
- Randomly sampled 20,000 rows for each sentiment category to balance the dataset.

- Combined the sampled data into a single DataFrame and shuffled it.

2. Data Encoding and Splitting

- Used Label Encoder to encode sentiments as 0, 1, and 2 for negative, neutral, and positive sentiments respectively.
- Split the data into 80% for training and 20% for testing.

3. Text Tokenization

- Tokenized texts using the `simple_preprocess` library.

4. Feature Extraction

- Extracted features using TF-IDF and Word2Vec embeddings.

5. Hyperparameter Tuning

- Created a random subset of 5,000 data points to find the best hyperparameters for each model using the `RandomizedSearchCV` library, as applying this on the complete dataset at once can be time-consuming for models like SVM.
- Tuned hyperparameters with 7-fold cross-validation for six ML models: SVM, Multinomial Naive Bayes, Gaussian Naive Bayes, XGBoost, Random Forest, and Logistic Regression.

6. Model Training

- After finding the best hyperparameters for each model, fitted them on the entire balanced training dataset.

7. Model Evaluation

- Plotted confusion matrix and classification report for each ML model, using both TF-IDF Vectorizer and Word2Vec embeddings.
- Saved all the models, loaded them, and applied them to separate app data after removing matching data from the earlier trained data to ensure all data were fresh for prediction.

8. Model Selection

- Saved the best-performing model for future sentiment prediction on new reviews, demonstrating the effectiveness of ML and DL models in sentiment analysis.

Observations

```
Accuracy using diff ML and word-embedding models in Combined App Data:
```

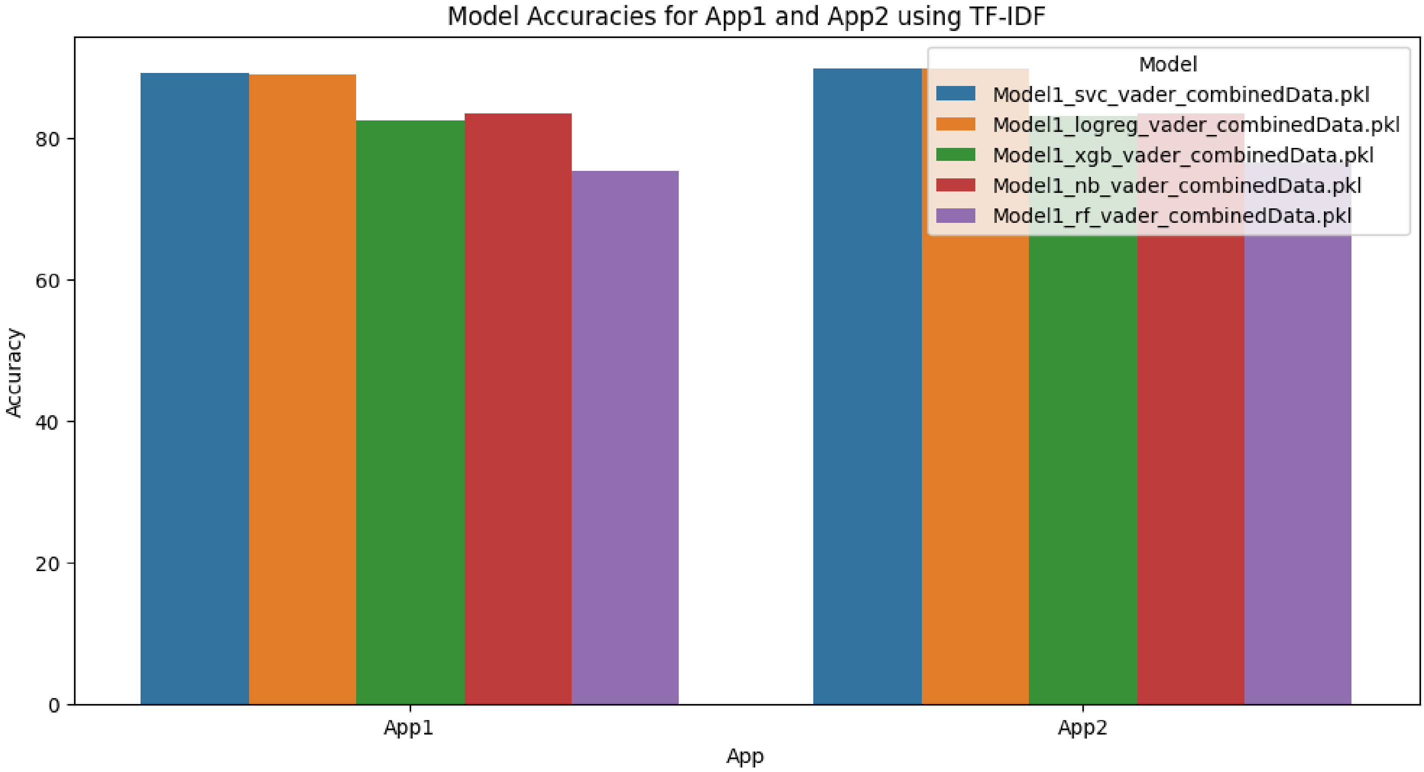
	Model	TF-IDF Accuracy (%)	Word2Vec Accuracy (%)
0	XGBoost	85.575000	73.133333
1	Logistic Regression	89.083333	60.966667
2	Random Forest	78.366667	69.566667
3	Naive Bayes	75.300000	56.750000
4	Support Vector Machines	90.525000	78.516667

```
Accuracy and Precision in App1 with different models using TF-IDF
```

	Model	Accuracy	Precision
4	Model1_svc_vader_combinedData.pkl	89.123081	0.933129
1	Model1_logreg_vader_combinedData.pkl	88.810825	0.931367
0	Model1_xgb_vader_combinedData.pkl	82.351028	0.920475
3	Model1_nb_vader_combinedData.pkl	83.463440	0.904025
2	Model1_rf_vader_combinedData.pkl	75.110591	0.902792

```
Accuracy and Precision in App2 with different models using TF-IDF
```

	Model	Accuracy	Precision
1	Model1_logreg_vader_combinedData.pkl	89.749006	0.930975
4	Model1_svc_vader_combinedData.pkl	89.749006	0.930645
0	Model1_xgb_vader_combinedData.pkl	83.001988	0.913437
2	Model1_rf_vader_combinedData.pkl	76.503479	0.897989
3	Model1_nb_vader_combinedData.pkl	83.449304	0.892447

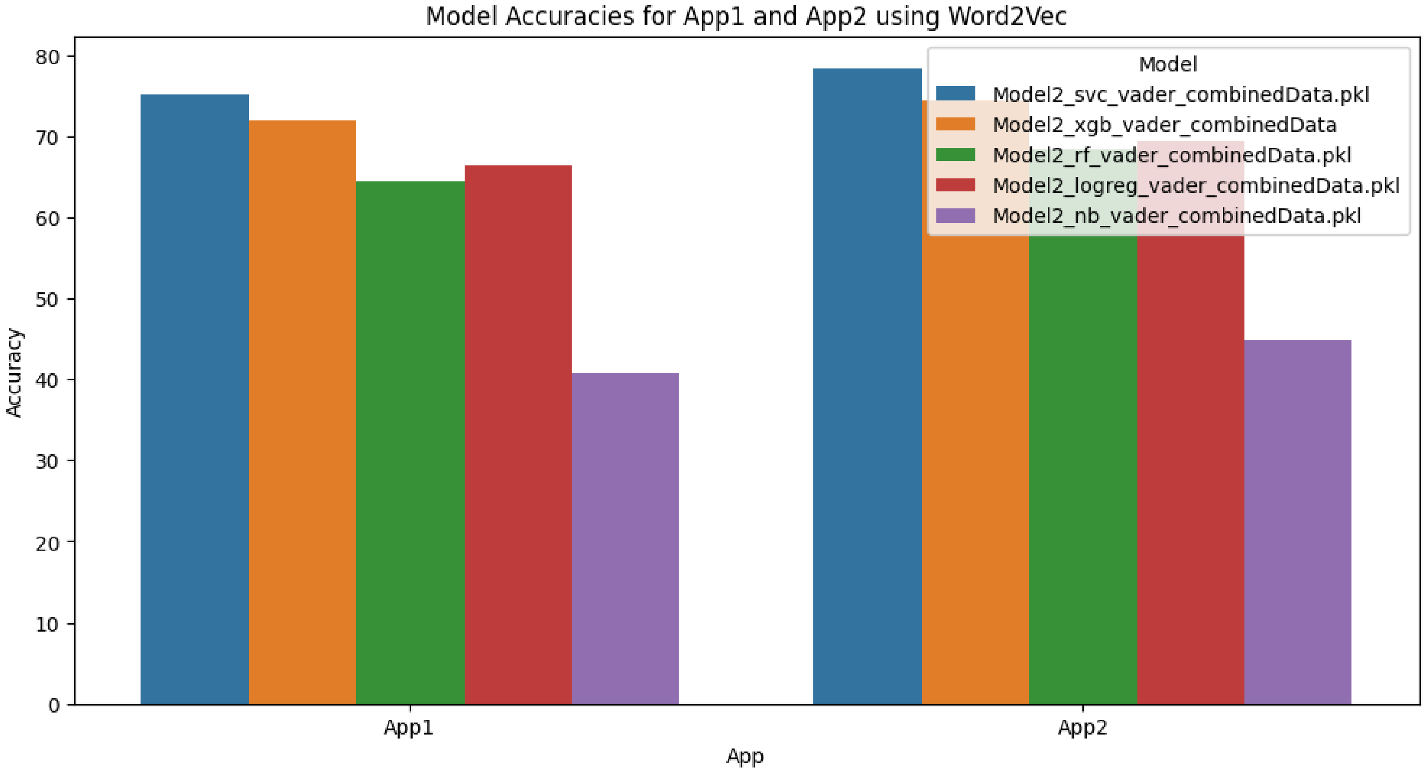



```
Accuracy and Precision in App1 with different models using Word2Vec
```

	Model	Accuracy	Precision
4	Model2_svc_vader_combinedData.pkl	75.052043	0.896363
0	Model2_xgb_vader_combinedData	71.864429	0.884974
2	Model2_rf_vader_combinedData.pkl	64.350768	0.876300
1	Model2_logreg_vader_combinedData.pkl	66.425969	0.873887
3	Model2_nb_vader_combinedData.pkl	40.671351	0.853722

```
Accuracy and Precision in App2 with different models using Word2Vec
```

	Model	Accuracy	Precision
4	Model2_svc_vader_combinedData.pkl	78.379722	0.891990
0	Model2_xgb_vader_combinedData	74.490557	0.876594
2	Model2_rf_vader_combinedData.pkl	68.414513	0.868240
1	Model2_logreg_vader_combinedData.pkl	69.371272	0.856291
3	Model2_nb_vader_combinedData.pkl	44.843439	0.832134



- The best-performing model was Logistic Regression, which achieved an accuracy of 90.85% and a precision of 0.92 on the complete combined app data.

DL Methodology

1. Data Preparation

- Followed all the steps of ML methodology till splitting data into train and test sets.

2. Data Padding

- Applied padding to the train and test data.

3. Model Application

- Applied DL models: CNN and Bidirectional LSTM.

4. Model Optimization

- Used callbacks for early stopping based on validation loss and created a model checkpoint to save the best model with the minimum validation loss.

5. Model Evaluation

- Plotted loss curve, accuracy curve, confusion matrix, and classification report.
- Predicted sentiments and applied the best model on test data to evaluate accuracy.

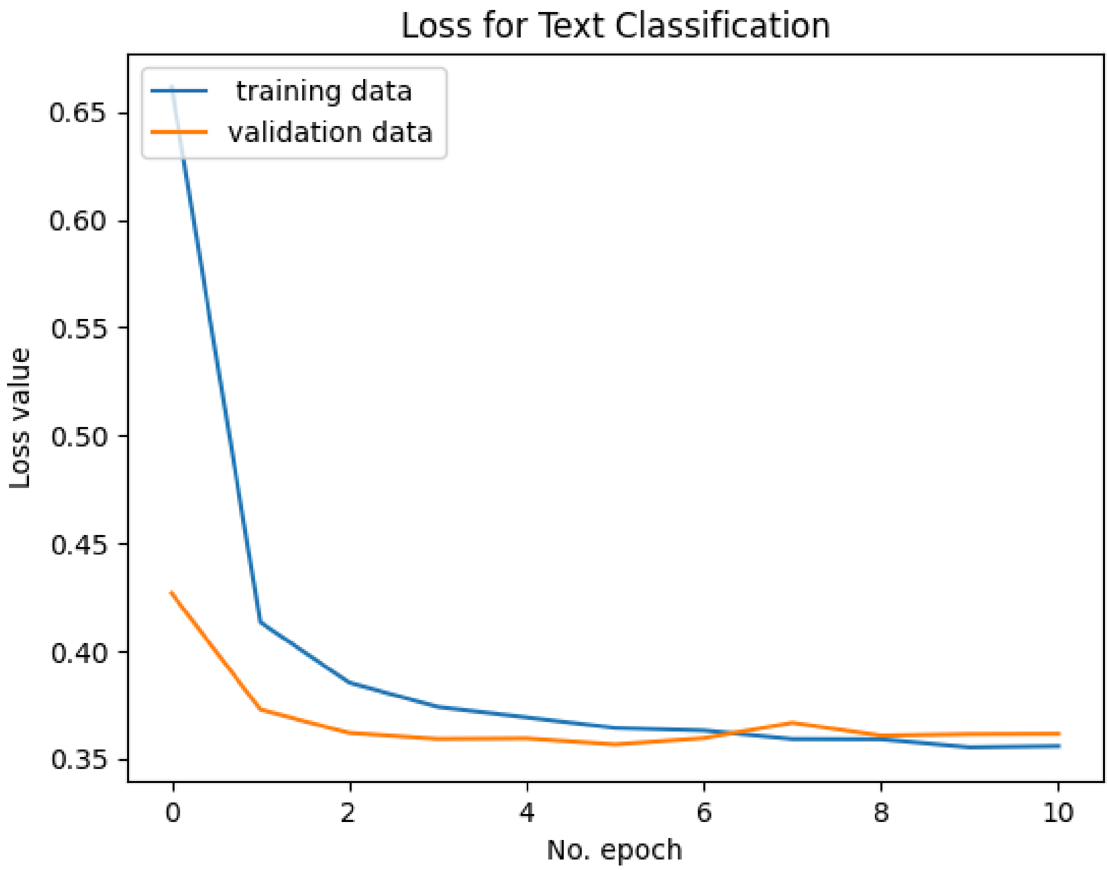
6. Model Deployment

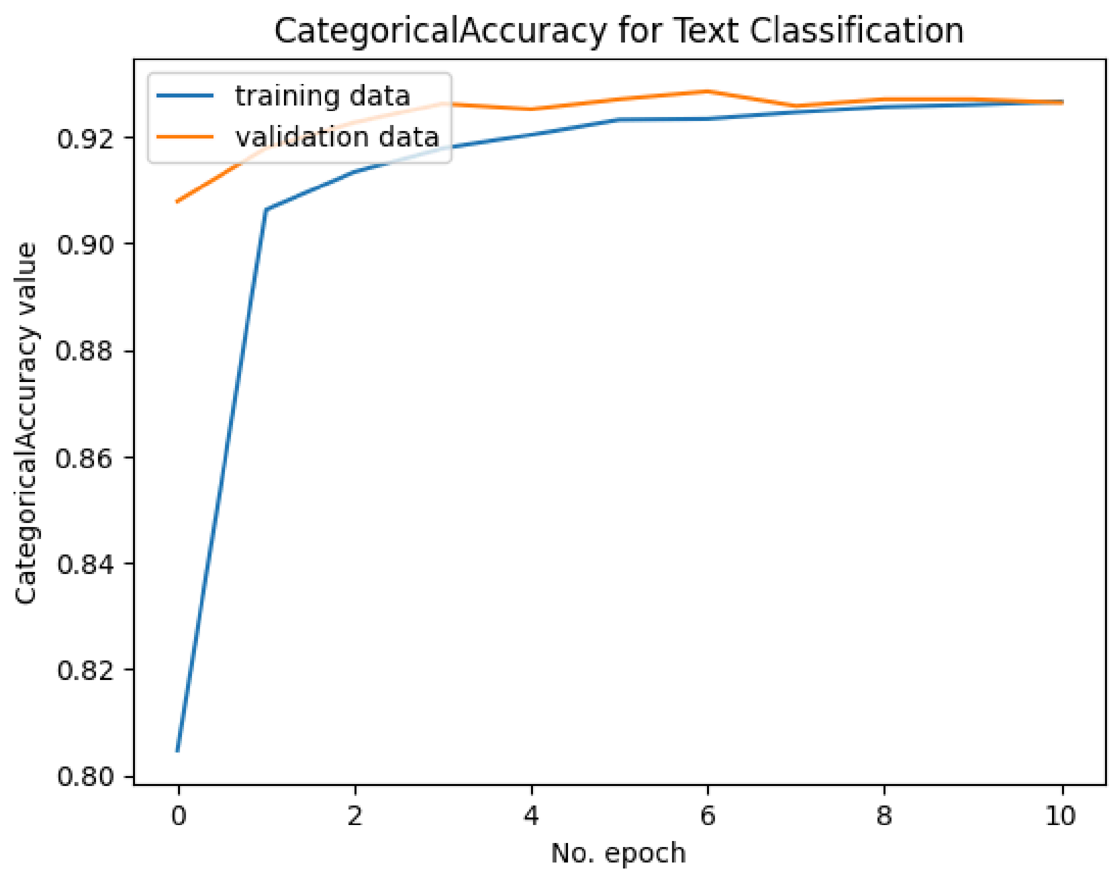
- Saved the best model, loaded it, and applied it to separate app data after removing matching data from the earlier trained data to ensure all data were fresh for prediction.

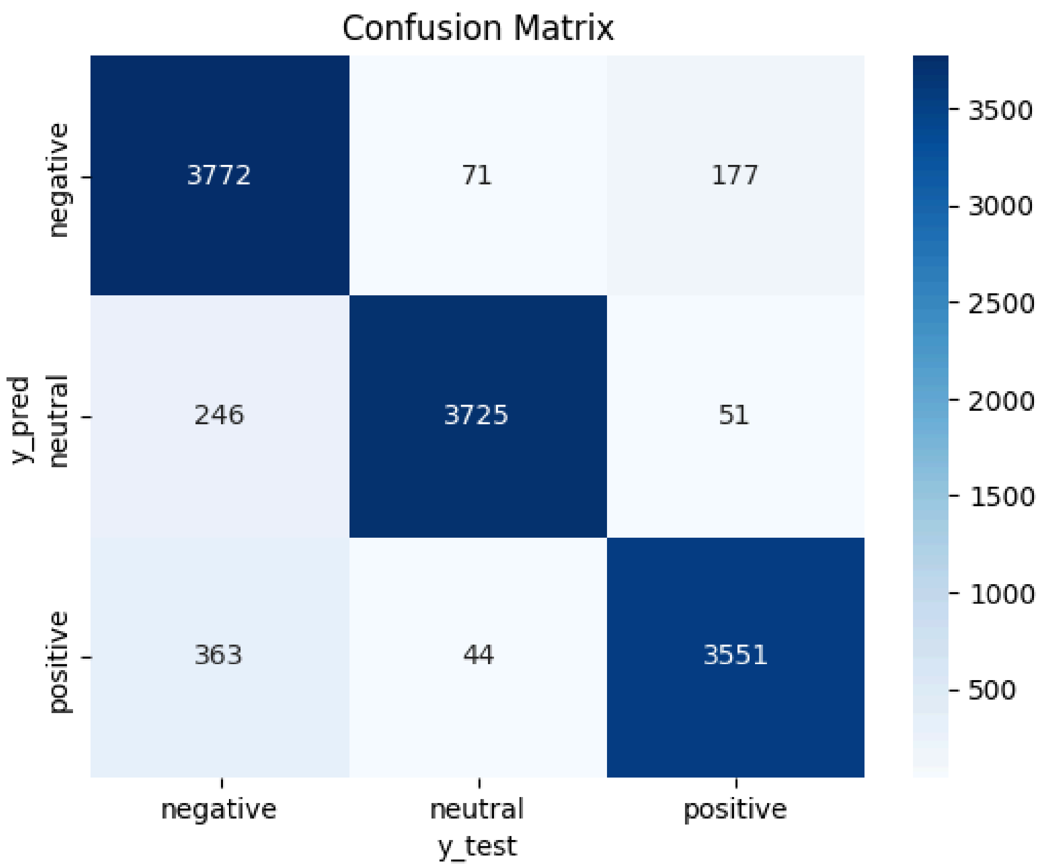
Observations

Model Performance

CNN Model:

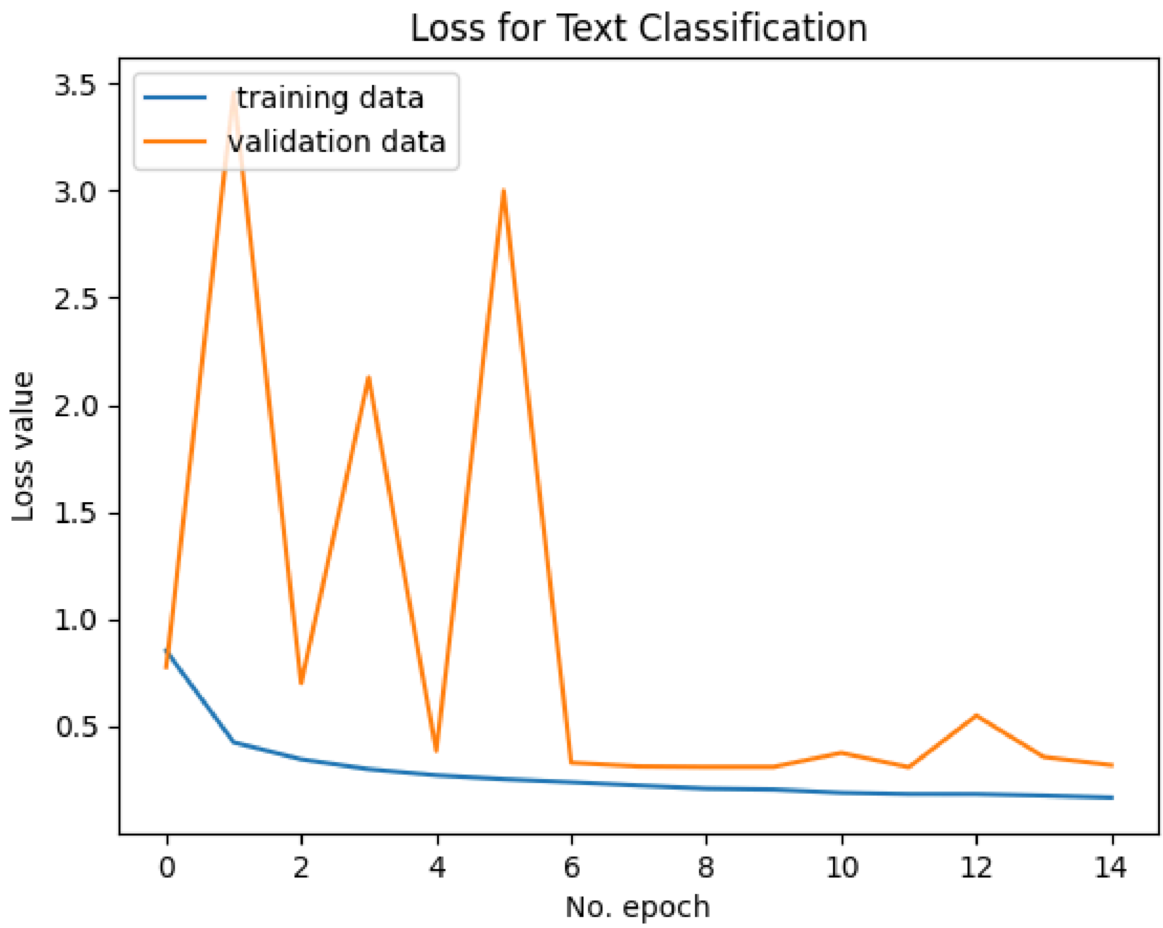




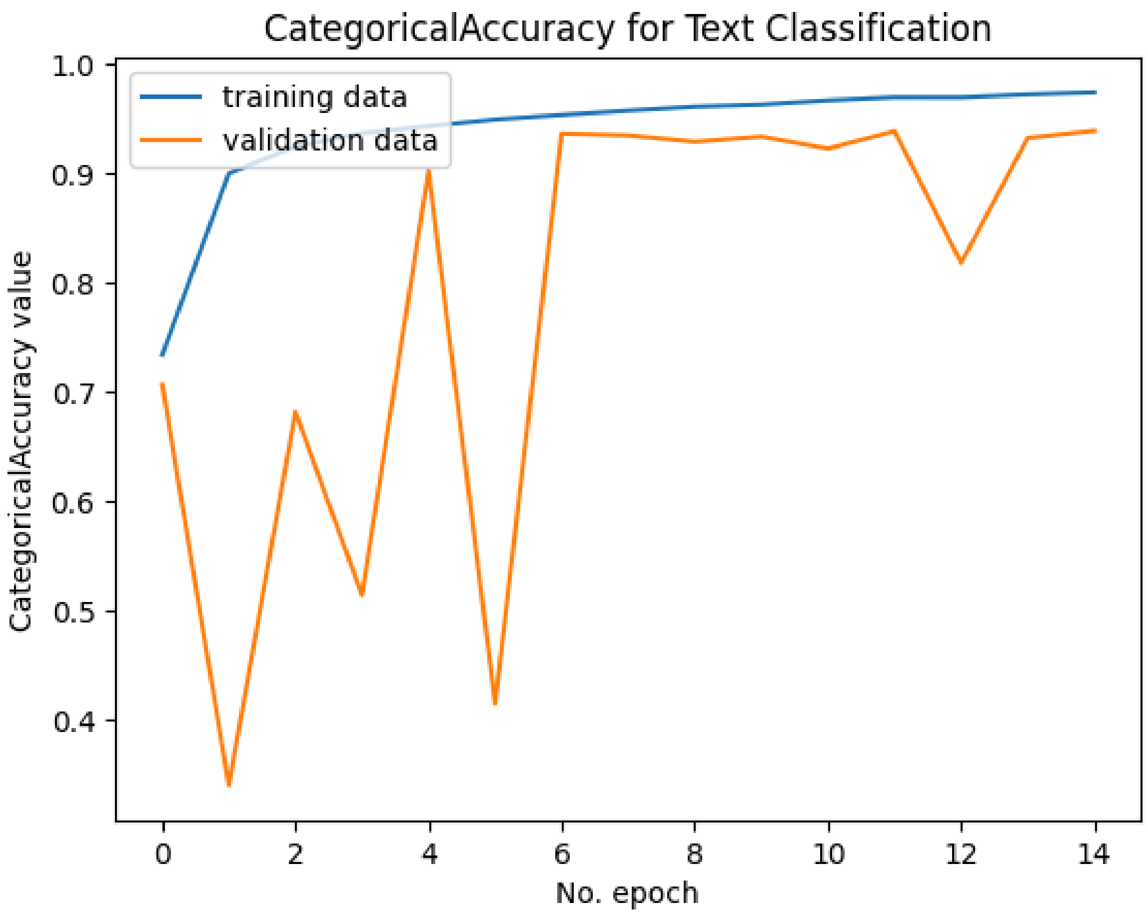


- Achieved an accuracy of 92.07% on balanced combined app data.
- Achieved an accuracy of 90.97% on app1 data.
- Achieved an accuracy of 91.77% on app2 data.

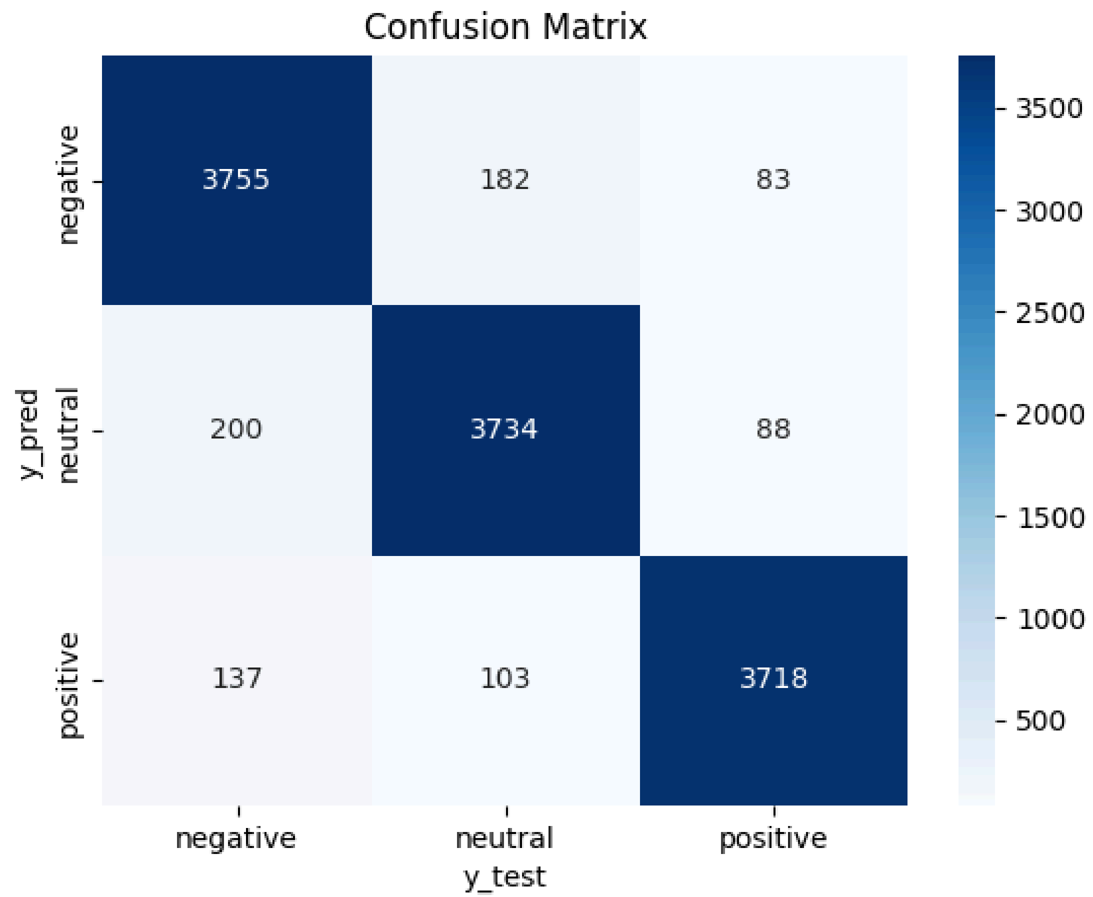
LSTM Model:



Loss for Text Classification
training data
validation data
Loss value
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
2
4
6
8
10
12
14
No. epoch



CategoricalAccuracy for Text Classification
training data
validation data
CategoricalAccuracy value
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
2
4
6
8
10
12
14
No. epoch



- Achieved an accuracy of 93.39% on balanced combined app data.
- Achieved an accuracy of 94.17% on app1 data.
- Achieved an accuracy of 94.71% on app2 data.